1979-12

Finite element analysis program (FEAP) for conduction heat transfer

Bettencourt, Jorge Martins

Monterey, California. Naval Postgraduate School

FINITE ELEMENT ANALYSIS PROGRAM (FEAP) FOR CONDUCTION HEAT TRANSFER

Jorge Martins Bettencourt

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

FINITE ELEMENT ANALYSIS PROGRAM (FEAP)

FOR CONDUCTION HEAT TRANSFER

by

Jorge Martins Bettencourt

December 1979

Thesis Advisor: G. Cantin

Approved for public release; distribution unlimited

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) Finite Element Analysis Program (FEAP) for Conduction Heat Transfer	5. TYPE OF REPORT & PERIOD COVERED Master's & Engineer's Thesis December 1979
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Jorge Martins BETTENCOURT	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940	12. REPORT DATE December 1979
	13. NUMBER OF PAGES 161
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Naval Postgraduate School Monterey, California 93940	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Conduction Heat Transfer, Finite Element, Numerical Analysis, Time Integration Operator, Computer Programming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Finite Element Analysis Program (FEAP) was expanded to solve linear and nonlinear, two and three dimensional heat conduction problems. The usual types of boundary conditions, including radiation, may be specified. A wide range of two- and three-time level schemes for the solution of time dependent problems is available in the program and a discussion of those most commonly used is presented. The algorithms for the solution of typical

practical problems are described and several numerical examples are presented. The results are compared with the available analytical solutions. A listing of this expanded version of FEAP and the corresponding user's instructions are provided.

Finite Element Analysis Program (FEAP) for Conduction Heat Transfer

by

Jorge Martins Bettencourt
Lieutenant, Portuguese Navy
B.S., Naval Postgraduate School, 1979

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

and

MECHANICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
December 1979

ABSTRACT

The Finite Element Analysis Program (FEAP) was expanded to solve linear and nonlinear, two and three dimensional heat conduction problems. The usual types of boundary conditions, including radiation, may be specified. A wide range of two- and three-time level schemes for the solution of time dependent problems is available in the program and a discussion of those most commonly used is presented. The algorithms for the solution of typical practical problems are described and several numerical examples are presented. The results are compared with the available analytical solutions. A listing of this expanded version of FEAP and the corresponding user's instructions are provided.

TABLE OF CONTENTS

ACKNOWLEDGMENTS

The author wishes to express his sincere appreciation to Professor Gilles Cantin for is invaluable guidance, help and friendship as instructor and thesis advisor. In addition, the author wishes to thank Professor Olgierd Cecil Zienkiewicz for his interest, comments and thoughtful advice.

The author is obligated to Professor Matthew Kelleher and Professor David Salinas for their assistance during the first stages of this work.

Finally the author wishes to thank his wife, Maria João, and his daughters, Joana and Catarina, for their understanding and encouragement throughout the course of this work.

I. INTRODUCTION

This thesis describes an application of the Finite Element method to Heat Transfer analysis through the use of a computer program. This program was designed to solve linear and nonlinear, steady and unsteady, two and three dimensional heat conduction problems involving temperature dependent thermophysical properties and complicated radiation/convection boundary conditions.

The Finite Element Analysis Program (FEAP), programmed by Professor R. L. Taylor in the Department of Civil Engineering of the University of California, Berkeley, served as the point of departure for the present code. It has now been expanded with two additional modules for Heat Transfer analysis and time integration of first order equations, respectively, but the original characteristics of a research and educational tool in which the various modules can be changed or added to as desired, were maintained.

Although all equations are retained in core, the program can handle realistic engineering problems with several hundred unknowns in most computer systems.

While the algorithms to solve steady heat transfer problems are well studied and defined, more research has to be done on the solution of unsteady problems. The Zienkiewicz two- and three-time level schemes were integrated in the program in such a way that a wide range of

choices of time integration algorithms is available to the user. A limited study was performed in order to determine the characteristics of the most used members of those families of numerical schemes.

The FEAP code is written in FORTRAN IV language and this version was constructed and tested on an IBM 360/67 computer system with OS/67 release 18. Other systems and installations will require modifications.

II. STATEMENT OF OBJECTIVES

A. EQUATION OF CONDUCTION. INITIAL AND BOUNDARY CONDITIONS

The problem considered for solution is thoroughly developed by Arpaci in Ref. 1 and is mathematicaly described in the region Ω by the equation

$$\rho\, c\, \frac{\partial T}{\partial t} = \overline{\nabla} \,.\, (\overline{k}\, \overline{\nabla} T) + Q \tag{1}$$

subjected to boundary conditions

$$T = T_w \qquad \text{on} \qquad \Gamma_1 \tag{2}$$

and

$$(\overline{k}\, \overline{\nabla} T)\, \overline{n} + q + q_c + q_r = 0 \qquad \text{on} \qquad \Gamma_2 \tag{3}$$

and initial condition

$$\lim_{t \to 0} T = T_0 \tag{4}$$

$\overline{\nabla}$ is the gradient operator, Γ_1 and Γ_2 are mutually exclusive parts of the boundary of the region Ω , T is the temperature and t is the time. The thermal capacity ρc, the thermal conductivity $\overline{k}$ and the rate of internal heat generation Q are thermophysical properties dependent on temperature.

In equation (3) $\overline{n}$ is an outward unit vector normal to the boundary surface while q, q_c, q_r represent respectively the imposed heat flux and the rates of heat flow per unit area due to convection and radiation defined as

$$q_c = h_c\, (T - T_{ac}) \tag{5}$$

and

$$q_r = F\sigma (T^4 - T_{ar}^4) = h_r (T - T_{ar}) \qquad (6)$$

In (5) h_c is the temperature dependent convective heat transfer coefficient. In (6) the parameter h_r is defined by the expression

$$h_r = F\sigma (T^2 + T_{ar}^2) (T + T_{ar}) ,$$ where F is the radiant exchange factor and σ the Stefan-Boltzman constant. T_{ac} and T_{ar} are the equilibrium temperatures for which, respectively, no convection and radiation occurs.

B. NUMERICAL SOLUTION. DISCRETIZATION BY THE GALERKIN METHOD. MATRIX FORMULATION

The spacewise discretization of equation (1) using cartesian coordinates can be acomplished by Galerkin's principle as shown by Zienkiewicz in Ref. 2 and Lew in Ref. 3.

Let the unknown function T be approximated, throughout the solution domain at any time t, by the relationship

$$T = \sum_{i=1}^{n} N_i(x,y,z)\ T_i(t) = \langle N\rangle \{T\} \qquad (7)$$

where N_i are the usual shape functions defined piecewise element by element, T_i being the nodal parameters.

The result is

$$[K]\ \{T\} + [C]\ \{\dot{T}\} + \{F\} = \{0\} \qquad (8)$$

where [K] and [C] are symmetric matrices defined, on the element level, as

$$[K]^e = \int_{\Omega^e} \Big(\langle\frac{\delta N}{\delta x}\rangle^T \langle\frac{\delta N}{\delta x}\rangle k_x + \langle\frac{\delta N}{\delta y}\rangle^T \langle\frac{\delta N}{\delta y}\rangle k_y + \langle\frac{\delta N}{\delta z}\rangle^T \langle\frac{\delta N}{\delta z}\rangle k_z \Big)\, d\Omega +$$
$$\int_{\Gamma_2^e} (h_c + h_r)\ \langle N\rangle^T \langle N\rangle\ d\Gamma \qquad (9)$$

$$[C]^e = \int_{\Omega^e} \rho c \langle N\rangle^T \langle N\rangle \, d\Omega \tag{10}$$

The load vector F is

$$\{F\}^e = -\int_{\Omega^e} \langle N\rangle^T Q \, d\Omega + \int_{\Gamma_2^e} \langle N\rangle^T (q - h_c T_{ac} - h_r T_{ar}) \, d\Gamma \tag{11}$$

where the surface integrals are performed only over the surfaces where the prescribed boundary condition applies.

It must be noted that the set of equations (8) is nonlinear since the matrices [K] , [C] and vector {F} are dependent on T .

The system of equations (8) may now be solved by any automated numerical technique. Code FEAP was written for this purpose and is described in the rest of this thesis.

III. FINITE ELEMENT ANALYSIS PROGRAM

The Finite Element Analysis Program (FEAP) was programmed and published by Taylor in Ref. 4 where it is well explained. The reader should consult this reference for details about it.

This thesis is an expansion of the original program towards the solution of Heat Conduction problems and as such only the main features of the basic program will be referred in this section. A more detailed explanation is reserved for the new subroutines added to the program. The complete user's instructions for FEAP are given in Appendix A and they complement this section.

A. MAIN PROGRAM

FEAP can be separated into two basic parts:

a) Data input module and preprocessor

b) Solution and output modules

The data input module must transmit sufficient information to the other modules so that each problem can be solved. All the input data is stored in a single array (integer array M) which is partitioned to store all the data arrays, as well as the global arrays, e.g., stiffness, mass, load, etc.

The total capacity of the program is controlled by the dimension of the array M in the blank common of the main

program and the corresponding value of the variable MAX.

The partition of M is performed in the control routine (subroutine PCONTR) and the data used for it is supplied by the user in the title and control information cards (first two cards for every run of the program).

Then the nodal coordinates, element connections, material properties, nodal loading and boundary conditions may be input using the macro control statements. They are interpreted by subroutine PMESH and each control is a function independent of the others.

To clarify what was said till this point, the discussion of a simple example may be useful. Consider the mesh presented in[1] Figure 1 and assume that quadratic rectangular elements as the one defined in Figure 2 are used. This example is completly solved in Appendix B and will be used throughout this text.

The sector of a circular ring was divided in six elements and the 33 nodes were numbered. The order of numbering is not crucial but in order to improve the profile of non-zero coefficients the following general rule should be used.

The numbering should be such as to minimize the nodal difference for each element (maximum node nummber minus minimum node number).

The user can now proceed to the preparation of data for the program. The first step consists of specifying the problem title and the control information. The latter is

[1]Figures are grouped at the end pp 47-79

clearly 33 nodes, six elements, one material set, two spatial dimensions and eight nodes per element. If this is a Heat Transfer problem we have one unknown per node (temperature).

The program expects now the data cards for mesh description. Any analysis require at least

a) coordinate data which follows the macro command COOR

b) connectivity table which follows the macro command ELEM

c) material data which follows the macro command MATE

In addition, most analysis will require specification of nodal boundary restraint conditions, macro BOUN, and the corresponding nodal force or displacement values, macro FORC. The term displacement refers to the value of the unknown for each specific problem. In a Heat Transfer problem the unknown is temperature while in an Elasticity analysis it is a geometrical displacement.

The TEMP macro command (temperature data) shall not be used in a Heat Transfer analysis to specify nodal temperature. It may be used to specify auxilary nodal quantities.

If in our problem we assume the line defined by nodes 31, 32 and 33 at the temperature of 20, the line defined by nodes 1, 2 and 3 at 820 and insulated elsewhere, the macro control statements and correspondent data cards will be

```
COOR
    1    5   .1
```

```
   31        .3
    4    5.116666667
   29       .283333333
    2    5   .1         3.
   32        .3         3.
    3    5   .1         6.
   33        .3         6.
    5    5.116666667  6.
   30       .283333333  6.
   (blank card)
POLA
    1   33     1
   (blank card)
ELEM
    1     1     1     6     8     3     4     7     5     2     5
   (blank card)
BOUN
    1     1    -1
    3          -1
   31     1    -1
   33          -1
   (blank card)
FORC
    1     1   820.
    3         820.
   31     1    20.
   33          20.
```

(blank card)

```
MATE
    1    2
10.      10.      8000.     250.                     3
END
```

In spite of the fact that the MATE command must be included in this part of the data preparation, its discussion is left to the section where the element modules are treated.

After completing the mesh data input, we are ready to initiate the problem solution.

FEAP has modules for variable algorithm capabilities and which, if necessary, can be modified or expanded. The basic aspect of the variable algorithm program is a macro instruction language which can be used to construct modules for specific algorithms as needed. This language is interpreted by subroutine PMACR and the complete list of macro instructions available in this version of FEAP is given in Appendix A.3.

Here, as an example, we will discuss in detail the algorithm for the solution of linear steady state problems using the original explanation given in Ref. 4.

To use the macro programming commands, the user only needs to learn the mnemonics of the language. If one wishes to form the global stiffness matrix the program instruction TANG is used (TANG is the mnemonic for a symmetric tangent stiffness matrix and for nonlinear elements would form and

assemble into the global stiffness the element tangent stiffness computed about the current displacement state; for linear elements this is just the linear stiffness matrix). For a problem with an unsymmetric tangent stiffness the macro command UTAN is used.

If one wishes to form the right hand side of the equations modified for specified displacements one uses the program instruction FORM. The resulting equations are solved using the instruction SOLV.

Printed output can be obtained using the macro command DISP.

The above instructions are sufficient to solve linear steady state problems, that is, the macro instructions

```
TANG ( or UTAN )
FORM
SOLV
DISP
```

are precisely the required instructions to solve any linear steady state problem.

If the same block of instructions is repeated twice or more, considerable effort is wasted in preparing the macro instruction data. To rectify this, looping commands are introduced as the instruction pair

```
LOOP      n
  .
  .
  .
NEXT
```

which indicates that looping over all instructions between

LOOP and NEXT will occur n times.

Many other classes of problems can be solved using the macro instruction list given in Appendix A. With the above short explanation and the discussion presented in section IV for Heat Transfer problems, the reader will be able to construct his own algorithms using the macro programming language.

B. FINITE ELEMENT SOLUTION MODULES

Most of FEAP is common to a wide range of problems, but some of the computations are linked to the type of problem to be solved. It is clear that if one wishes to solve an Elasticity problem, the calculation of the stiffness matrix will be different from that used for a Heat Transfer problem. It is also clear that differences also arise if one is using triangular elements instead of quadratic isoparametric elements as a way of describing the continuum.

The program was designed such that all computations associated with any type of element are contained in an element subroutine called ELMTnn where nn is between 01 and 05 in this version of FEAP. Each element type to be used is specified as part of the material property data, following macro control MATE.

Since the objective of this work is the solution of Heat Transfer problems, the element modules discussed here are the two and three dimensional heat tranfer modules named ELMT02 and ELMT03.

1. Two Dimensional Heat Transfer Element

Subroutine ELMT02 is accessed using the number two in column ten of each material number card.

According to the value of the variable ISW, defined in subroutine PMACR, a specific function is performed. If ISW=1 it reads and prints the material property data and line boundary conditions. The stiffnes and mass matrices are computed if ISW=3 and ISW=5 respectively. The load vector due to internal forces and boundary conditions is calculated when ISW=6.

The material properties may be input as constants or as a table of temperature and property values. If any other way of defining the property values is required by a specific problem, subroutines PKX , PKY , PROC or PQ which calculate respectively the conductivity in x and y directions, the thermal capacity and the heat generation per unit volume, can be easily modified or substituted.

The shape functions used by this heat tranfer module are calculated by subroutines SHAPE and SHAP2. These routines are capable of constructing shape functions for a three-node triangle, a four-node linear quadrilateral, an eight-node quadratic serendipity quadrilateral, a nine-node quadratic Lagrangian quadrilateral or any combination in-between.

A four- to nine-node two dimensional element is shown in Figure 3. As shown by Bathe and Wilson in Ref. 5 the temperature within the element is expressed at any time

in the local coordinate system s, t in terms of the nodal temperatures T_i by

$$T(s,t) = \Sigma\ N_i(s,t)\ T_i$$

where

$$N_1 = (1-s)(1-t)/4 - (N_5+N_8)/2 - N_9/4$$
$$N_2 = (1+s)(1-t)/4 - (N_5+N_6)/2 - N_9/4$$
$$N_3 = (1+s)(1+t)/4 - (N_7+N_6)/2 - N_9/4$$
$$N_4 = (1-s)(1+t)/4 - (N_7+N_8)/2 - N_9/4$$
$$N_5 = (1-s^2)(1-t)/2$$
$$N_6 = (1+s)(1-t^2)/2$$
$$N_7 = (1-s^2)(1+t)/2$$
$$N_8 = (1-s)(1-t^2)/2$$
$$N_9 = (1-s^2)(1-t^2)$$

If any of the nodes from five to nine are omitted the corresponding value of N is zero.

The connectivity table must follow the numbering convention shown in Figure 3, otherwise wrong results will be obtained.

The maximum number of nodes per element will depend on the type of elements to be used and may be from three to nine. The eight-node serendipity shape functions occur if the ninth node number is omitted. The four-node quadrilateral shape functions are computed if only the first four nodal connections are non zero and the three-node triangle if the first three nodal connections are non zero. Finally, if a mid-side nodal connection is omitted the edge is linear.

The line boundary conditions, closely related to the type of element used, are defined in the cards following the macro MATE. The line subjected to a specified boundary condition is identified using the local coordinates. A line is numbered 1 or 2 if it is perpendicular to the axis s or t, respectively. Those numbers are positive or negative according to which direction of the axis it is perpendicular. Thus, as an example, the line defined by the nodes 2, 6 an 3 is numbered 1 while the line defined by nodes 1, 5 and 2 is numbered -2.

The boundary conditions are treated in subroutine BCOND2 and four types are considered:

a) specified flux

b) convection with constant heat tranfer coefficient

c) convection with temperature dependent heat transfer coefficient

d) radiation

If more complex boundary conditions are required subroutine BCOND2 may be easily modified.

The numerical integration necessary to evaluate the matrices and vectors in equation (6) are performed using the Gauss quadrature formula; the abscissas and weight coefficients are calculated in subroutine PGAUSS. The user may choose the number of points per direction used in the integration from one to six, but the default value is four points .

To illustrate the above, we assume that our sector

used as an example is part of the cross section of a hollow cylinder made of a material with conductivity in x and y directions of 10, specific heat 250 and density 8000. If the outside surface is subjected to convection to an ambient atmosphere at 20 degrees with a constant heat tranfer coefficient of 200, the cards following the macro card MATE are

```
MATE
    1    2
10.       10.       8000.     250.                 3
    6    2    1  200.      20.
```

It must be noted that a plane analysis is specified and the integration will be performed using three points per direction. The reader must also note that this version of the problem is slightly different from the one given when the mesh data preparation was discussed. In the previous problem the line now subjected to a convection boundary condition was at a specified constant temperature of 20. If this second problem is to be solved the cards corresponding to nodes 31-33 in the BOUN and FORC sets must be omitted.

Following is a list of subroutines included in the two dimensional heat tranfer module:

ELMT02 : main routine; forms the necessary arrays for the solution.

SHAPE : calculates the shape functions for the triangle and the linear quadrilateral elements.

SHAP2 : adds the quadratic terms and the center node

to the shape functions.

PGAUSS : determine the abscissae and weight coefficients for the numerical integration.

BCOND2 : adds to the arrays calculated in ELMT02 the contribuition from the specified boundary conditions.

PKX : determine the temperature dependent conductivity in the x direction.

PKY : determine the temperature dependent conductivity in the y direction.

PROC : determine the temperature dependent thermal capacity.

PQ : determine the temperature dependent heat generation per unit volume.

CONV : determine the temperature dependent heat transfer coefficient.

TABLE : called by PKX, PKY, PQ, PROC and CONV; calculates the correspondent coefficient to a given temperature by linear interpolation between two consecutive entries in a table.

JACBB2 : calculates the jacobian determinant when an integration over a line is necessary.

2. Three Dimensional Heat Transfer Element

This module is called ELMT03 and is accessed using the number three in column ten of each material number card.

This element is the generalization of ELMT02 for three dimensional space and little remains to be said about its use.

The material properties are read in a similar way as the two dimensional case, but the conductivity in the z direction must be added. This module uses the same subroutines as ELMT02 to read and calculate the temperature dependent properties.

Subroutines SHAP3D and SHAP3 may construct interpolation functions for any combination between a eight-node linear brick and a 21-node Lagrangian brick.

An eight- to twenty one-node three dimensional solid element is shown in Figure 4. The natural coordinates (r, s and t) of the eight corner nodes are (± 1, ± 1, ± 1), of the twelve mid-edge nodes are (0, ± 1, ± 1), (± 1, 0, ± 1) and (± 1, ± 1, 0) and of the center node are (0, 0, 0).

The temperature within the element is defined in terms of the nodal temperatures T_i at any time by [Ref. 5]

$$T(r,s,t) = \Sigma\, N_i(r,s,t)T_i$$

If we define

$$G(\beta,\beta_i) = .5(1+\beta_i\beta) \quad , \quad \text{for } \beta_i = \pm 1$$
$$= 1-\beta^2 \qquad \text{for } \beta_i = 0$$

and

$$g_i = G(r,r_i)G(s,s_i)G(t,t_i)$$

where r_i, s_i and t_i are the natural coordinates of the element nodal points, the interpolation functions N_i are

$$N_1 = g_1 - (N_9 + N_{12} + N_{17})/2 - N_{21}/8$$
$$N_2 = g_2 - (N_9 + N_{10} + N_{18})/2 - N_{21}/8$$
$$N_3 = g_3 - (N_{10} + N_{11} + N_{19})/2 - N_{21}/8$$
$$N_4 = g_4 - (N_{11} + N_{12} + N_{20})/2 - N_{21}/8$$

$$N_5 = g_5 - (N_{13}+N_{16}+N_{17})/2 - N_{21}/8$$
$$N_6 = g_6 - (N_{13}+N_{14}+N_{18})/2 - N_{21}/8$$
$$N_7 = g_7 - (N_{14}+N_{15}+N_{19})/2 - N_{21}/8$$
$$N_8 = g_8 - (N_{15}+N_{16}+N_{20})/2 - N_{21}/8$$
$$N_j = g_j - N_{21}/4 \qquad \text{for } j=9,\ldots,20$$
$$N_{21} = g_{21}$$

If any of the nodes from nine to twenty one are omitted the corresponding value of g is zero.

The numbering convention used in the definition of the shape functions and shown in Figure 4 must be followed by the user when the connectivity table is built and the surface boundary conditions are specified. The surfaces are numbered ± 1, ± 2, ± 3 as they are perpendicular to the positive or negative directions of the axis r, s, t, respectively.

As the numerical integration routine is the same as for ELMT02, the user may choose from one to six points per direction, the default value being four points.

Following is a list of subroutines included in the three dimensional heat transfer module:

ELMT03 : main routine; forms the matrices necessary for the solution.

SHAP3D : calculates the shape functions for the eight node linear brick.

SHAP3 : adds the quadratic terms to the shape functions and the center node for the Lagrangian brick.

BCOND3 : adds to the matrices the contribuition from

the specified boundary conditions.

PKZ : determine the conductivity in the z direction

JACBB3 : calculates the jacobian determinant when an integration over a surface is necessary.

C. TIME INTEGRATION MODULE

The first order ordinary differential equation solver is accessed when the macro command ODE1 is utilized and uses the Zienkiewicz two- and three-time level schemes. The details of these algorithms are treated in section IV.

This module was designed for the solution of first order equations but can, with little programming effort, be expanded in order to include higher order algorithms and solve higher order differential equations.

In order to economize computer memory space at some cost of execution time, this module only uses the space previously reserved for the mesh construction. As the mesh data is not needed during the execution of a time step integration, all the corresponding arrays are kept in file 9 during that period. Once the next displacement vector is calculated and the current displacement vector is saved in file 10, all the information in file 9 is retrieved and the mesh data restored.

File 10 is used as a working space during the time integration and the current displacement vector is saved in it for possible future use. This vector is retrieved if the algorithm used in the next time step integration is a three point scheme.

Besides the time step size change using the appropriate macro instructions (macro TOL), an optional automatic time step adjustment was incorporated in the program. The norm of the difference between the displacements vectors at two consecutive times is computed at each step. If the norm is less than a predetermined value ΔT_{max}, the time step size is doubled before going to the next step, whereas if the norm is greater than a prespecified value ΔT_{min}, the time step size is halved and the calculation for that time step is repeated until the norm is acceptable. The magnitudes of the maximum and minimum values of the norm are problem dependent and must be choosen by the user. If they are not specified no time step adjustment will be performed. The recalculation of the displacement vector is allways done using the two point scheme, even when a three point scheme was utilized for the first calculation.

The macro command ODE1 used to access this module must be followed by a second macro command in order to determine the function to be performed. The secondary macro instructions are INIT, LINE or QUAD.

The couple ODE1 INIT reads the integration constants theta , beta and gamma, the parameters for the automatic time step adjustment and the initial displacement vector. This data must follow the macro program (see user's instructions). No time integration is performed by this instruction.

The couple ODE1 LINE performs the two point scheme

and the current displacement vector is substituted by the new calculated displacement vector.

The couple ODE1 QUAD has a function similar to ODE1 LINE but uses the three point scheme.

Once the execution of an ODE1 macro command is complete, the mesh data is restored and the displacement at time t replaced by the displacement at time $t+\Delta t$, when the secondary macro command is LINE or QUAD. If INIT is used, the displacement vector becomes the given initial displacement vector.

The subroutines included in this module are:

PODE1 : control routine; reserves working space and calls the appropriate subroutines.

INIT : reads the constants theta, beta and gamma and the maximum and minimum values for the norm used in the time step adjustment.

PLINE : performs the two point integration scheme.

PQUAD : performs the three point integration scheme.

TERM : performs the automatic time step adjustment.

IV. APPLICATION OF FEAP TO HEAT TRANFER PROBLEMS

The program FEAP requires from the user the knowledge of the algorithm to be used in the solution of the problem to be treated.

In this section the algorithms used in the solution of heat conduction problems are discussed.

A. STEADY STATE PROBLEMS

1. Algorithms for Linear Problems with Macro Program

In this case equation (8) becomes

$$[K]\ \{T\} + \{F\} = \{0\} \tag{12}$$

The algorithm used to solve (12) is described by

$$[K]\ \{v\} = -\{F\} - [K]\ \{T^\circ\} \tag{13}$$

$$\{T\} = \{T^\circ\} + \{v\} \tag{14}$$

where $\{T^\circ\}$ are the specified nodal temperatures.

The macro instruction TANG builds the left hand side of (13) while FORM forms the vector in the right hand side. The instruction SOLV solves the system (13) and performs the step defined by (14).

Consequently the simplest macro program used to solve this problem is

```
TANG
FORM
SOLV
DISP
```

2. Algorithms for Nonlinear Problems with Macro Program

Equation (12) still applies to this case but now [K] and {F} may be dependent on the nodal temperatures {T}. The well known Newton-Raphson iteration may be used and the algorithm is described as

$$[K(T^i)]\ \{v^i\} = -\{F(T^i)\} - [K(T^i)]\ \{T^i\} = \{R^i\} \qquad (15)$$

$$\{T^{i+1}\} = \{T^i\} + \{v^i\} \qquad (16)$$

where $\{T^i\}$ are the nodal temperatures at the ith iteration. The algorithm defined by (15) and (16) is the one used for the linear problem repeated several times. Then the macro program may be

```
LOOP    n
TANG
FORM
SOLV
DISP
NEXT
DISP
```

where n are the user's guess of the number of iterations necessary to obtain equilibrium. However, the program has an internal check on the value of the vector norm $||R^i||$, where

$$||R|| = \left(\sum_{k=1}^{n} |R_k^i|^2\right)^{1/2}$$

Whenever

$$||R^i|| < TOL \times \max_j ||R^j|| \quad (j=1,\ldots,i)$$

where TOL is a predefined tolerance (10^{-9} is the default

value), the iteration ceases and a skip to the macro command immediately following the first NEXT occurs. Usually one begins with zero as the initial guess of the displacement vector; however, any other vector may be used.

B. TIME DEPENDENT PROBLEMS

1. Two and Three Point Recurrence Schemes for First Order Equation

The set of differential equations

$$[K]\{T\} + [C]\{\dot{T}\} + \{F\} = \{0\} \tag{8}$$

may be solved using one of the many recursive schemes described in the literature. From them, the two and three time level schemes presented by Zienkiewicz [Ref.4] offer a wide range of choices for the solution of linear and nonlinear problems.

The recurrence relation for the two-time level scheme can be written as

$$\left(\frac{1}{\Delta t}[C] + \Theta[K]\right)\{T_{n+1}\} + \left(-\frac{1}{\Delta t}[C] + (1-\Theta)[K]\right)\{T_n\} + \{\overline{F}\} = \{0\} \quad , \quad 0 \leq \Theta \leq 1 \tag{17}$$

where the subscripts n and n+1 denote evaluation at time t and t+Δt. The vector $\{\overline{F}\}$ is defined as

$$\{\overline{F}\} = \{F_{n+1}\} + (1-\Theta)\{F_n\} \tag{18}$$

The choice of the parameter Θ defines the particular scheme to be used and the reader will recognize a well known series of finite difference formulas with a modification of using a weighted loading term $\{\overline{F}\}$.

Consider the system of decoupled equations in terms of the modal participation variables T_i

$$c_i \dot{T}_i + k_i T_i + f_i = 0 \qquad (19)$$

for free response, i.e., $f_i=0$, expression (17) becomes

$$(\frac{1}{\Delta t} c_i + k_i\Theta) (T_i)_{n+1} + (-\frac{1}{\Delta t} c_i + k_i(1-\Theta)) (T_i)_n=0 \qquad (20)$$

Substituting the relation

$$(T_i)_{n+1} = \lambda(T_i)_n \qquad (21)$$

in (20), the resulting characteristic equation of the recurrent scheme solved for λ gives

$$\lambda = [1-k_i(1-\Theta)\Delta t/c_i]/[1+k_i\Theta \Delta t/c_i]$$

The scheme will be unconditionally stable if

$$|\lambda| < 1$$

for any value of p_i where

$$p_i = (k_i/c_i) \Delta t \qquad (22)$$

This condition is satisfied for

$$\Theta \geq 1/2$$

On the other hand if

$$0 < \Theta < 1/2$$

stability is conditional requiring

$$p_i < 2/(1-2\Theta)$$

Figure 5 shows how λ varies with p_i for the schemes discussed later and how it compares with the exact value of

$$\lambda = \exp(-p_i)$$

The schemes considered are

a) Crank-Nicolson with Θ=1/2

b) Θ=2/3 proposed by Zienkiewicz [Ref. 4]

c) Θ=3/4 proposed by the author

d) Θ=0.878 proposed by Liniger in Ref. 6.

The Zienkiewicz three-time level scheme is defined

as

$$(\gamma[C] + \beta\, \Delta t[K])\{T_{n+1}\} + ((1-2\gamma)[C] + (1/2-2\beta+\gamma)\Delta t[K])\{T_n\} +$$
$$(-(1-\gamma)[C] + (1/2+\beta-\gamma)\Delta t[K])\{T_{n-1}\} + \Delta t\{F\} = \{0\} \qquad (23)$$

where the subscripts n, n+1 and n-1 denote evaluation at time t, $t+\Delta t$ and $t-\Delta t$ respectively.

The vector $\{\bar{F}\}$ is defined as

$$\{\bar{F}\} = \beta\{F_{n+1}\} + (1/2-2\beta+\gamma)\{F_n\} + (1/2+\beta-\gamma)\{F_{n-1}\} \qquad (24)$$

and the parameters γ and β define the particular scheme to be used.

Considering again the system of decoupled equations (19) and using the relations (21) and (22) one finds that the characteristic equation for the three-time level scheme is

$$(\gamma+\beta p_i)\, \lambda^2 + [(1-2\gamma) + (1/2-2\beta+\gamma)p_i]\lambda +$$
$$[-(1-\gamma) + (1/2+\beta-\gamma)p_i] = 0 \qquad (25)$$

Writing

$$g = [1+(1/2+\gamma)p_i]/[\gamma+\beta p_i]$$
$$h = [-1+(1/2-\gamma)p_i]/[\gamma+\beta p_i]$$

the roots of (25) may be written as

$$\lambda_{1,2} = (2-g)/2 \pm [(2-g)^2-4(1-h)]^{1/2}/2$$

These roots will be complex if the quantity under the square root is negative. Then the modulus of λ is

$$|\lambda| = (1-h)^{1/2}$$

The scheme is stable if $|\lambda|<1$ for any p_i and Wood in Ref. 7 shows that this condition is satisfied if

$$\gamma>1/2 \qquad \text{and} \qquad \beta>\gamma/2$$

Equation (25) has two roots: λ_2, the principal root which is the aproximation to the exact

$$\lambda = \exp(-p_i)$$

and the spurious root λ_2. For relative stability one must have

$$|\lambda_1| < |\lambda_2|$$

Also a negative root or complex roots can produce oscillation.

In Figure 6a-f the real roots λ_1 and λ_2 and the modulus $|\lambda|$ for the complex roots are plotted against p_i for the schemes:

a) β=1/3, γ=1/2 proposed by Lees in Ref. 8

b) β=3/4, γ=1 proposed by Hogge in Ref.9

c) β=0.646, γ=1.2184 proposed by Wood [Ref. 7]

d) β=4/5, γ=3/2 proposed by Zienkiewicz [Ref. 4]

e) β=9/10, γ=3/2 proposed by the author

f) Fully implicit algorithm, β=1, γ=3/2 [Ref. 7].

From Figure 6a one may predict a very strong oscillatory behaviour for the Lees algorithm a).

In order to study the various schemes, the physical problem represented by the nondimensionalized linear heat conduction equation

$$\frac{\delta T}{\delta t} = \frac{\delta^2 T}{\delta x^2}$$

was solved in a bar of length 4 and width 1 subjected to the boundary conditions

$$T=1 \quad \text{at} \quad x=0$$

$$\frac{\delta T}{\delta x}=0 \quad \text{at} \quad x=4$$

and initial condition

$$T=0 \quad \text{at} \quad t=0$$

i.e., consider a step change in the surface temperature. This problem was also studied by Wood and Lewis in Ref. 10.

The problem was discretized in space using ten equal length two dimensional quadratic elements through the length of the bar. This type of elements was chosen because it was reported by Wood and Lewis as giving the worst numerical results.

In order to avoid the discontinuity in the loading term caused by the step change of the surface temperature at the initial time, the problem was first solved starting from time $t=\Delta t$, where Δt is the time step size. The value of the temperature distribution at $t=\Delta t$ is provided by the exact analytical solution. This procedure also provides the necessary two starting vectors for the three-time level schemes.

The temperature of the nodes at $x=1$ is used as reference value to compare the various schemes.

In Figure 7a-d the results obtained from the various two-time level schemes with $\Delta t=2$ for the temperature at $x=1$ are compared with the analytical solution. For this ideal starting conditions the Crank-Nicolson, $\theta=1/2$, proves to be the most accurate algorithm. When the step size was increased to 10, all the schemes performed well as shown in Figure 8a-d.

The corresponding results for the three-time level

schemes are presented in Figure 9a-f and Figure 10a-f for $\Delta t=2$ and $\Delta t=10$, respectively.

Strong oscilations were observed with the Lees algorithm a).

For $\Delta t=2$ the schemes c), d) and e) performed well. For the larger step size of 10 the safest schemes are e) and f).

The step change in the loading term has been reported [Ref. 10] as producing relative instability when some of the schemes are used, particularly with the Crank-Nicolson algorithm. The noise may be supressed reducing the magnitude of the time step but this is impraticable in most problems. Other techniques may be used to reduce the amplitude of the oscillations as those discussed by Wood and Lewis [Ref. 10] and Gresho and Lee in Ref. 11.

The step change in the loading term in the problem in study is due to the variation of the surface temperature at the initial time t=0. That discontinuity in the force term is illustrated in Figure 11a. When the time dimension is discretized it has been common practice to calculate the numerical solution at the nodes t=0, $t=\Delta t$, etc. When a three-time level scheme is used the node at $t=-\Delta t$ is considered assuming all conditions steady before t=0. The force values used by this method are indicated by the circles in Figure 11a. This procedure transfers to the numerical solution the uncertainty in the value of the force

term at time t=0 introduced by the mathematical discontinuity assumed in the analytical solution. This problem is avoided if one uses the procedure illustrated in Figure 11b. The nodes considered for the discretization are those at $t=-\Delta t/2$, $t=\Delta t/2$, $t=3\Delta t/2$, etc. The discontinuity at t=0 is smoothed by the interpolation of the loading term inherent to the particular scheme being used. The values of the force term for every node are well defined and no uncertainty is associated with any of them. For the three-time level schemes the node at $t=-3\Delta t/2$ is used assuming all conditions steady before t=0.

The problem studied was solved using both starting procedures. The results obtained with the first procedure, start at t=0, using the two- and three-time level schemes are shown in Figures 12a-d and 13a-f, respectively. The second procedure, start at $t=-\Delta t/2$, gives the results shown in Figures 14a-d and 15a-f for the two- and three-time level schemes, respectively.

One may conclude that, in this simple problem, when the procedure illustrated in Figure 11b is followed, the step change in the loading term do not deteriorate the accuracy of the numerical results obtained with the schemes tested, with the exception of the Crank-Nicolson algorithm. This scheme shows an oscillatory behaviour not observed when the step change in the force term is not present.

The starting procedure of Figure 11b also proves to be an efficient way of starting the time integration with

the three-time level schemes.

2. <u>Algorithms for Time Dependent Problems with Macro Program</u>

A simplified form of the two-time level scheme is available in FEAP and is described by

$$(\frac{1}{\Delta t}[C_n] + \theta[K_n])\{T_{n+1}\} + (-\frac{1}{\Delta t}[C_n] + (1+\theta)[K_n])\{T_n\}+$$
$$\Delta t\{F\} = \{0\} \qquad (26)$$

where the subscripts n and n+1 denote evaluation at time t and t+Δt, respectively.

The three-time level scheme programmed in FEAP is described by the expression

$$(\gamma[C_n] +\beta\ \Delta t[K_n])\{T_{n+1}\} + ((1-2\gamma)[C_n] + (1/2-2\beta+\gamma)\Delta t[K_n])\{T_n\}$$
$$+ (-(1-\gamma)[C_n] + (1/2+\beta-\gamma)\Delta t[K_n])\{T_{n-1}\}$$
$$+ \{F_n\} = \{0\} \qquad (27)$$

where the subscripts n, n+1 and n-1 denote evaluation at times t,Δt+ t and t-Δt respectively.

It must be noted that in (26) and (27) the vector $\{F_n\}$is an aproximation of the interpolated force vector $\{\overline{F}\}$.

If the force vector is strongly dependent on time, this aproximation may produce wrong results.

In problems where the stiffness matrix is constant and the force term results from the specified boundary displacements or where the specified boundary forces are only time dependent, the interpolation of the force term may be easily acomplished since the force/displacement boudary values can be changed at any time using the macro commands MESH or PROP. This procedure was found partcularly useful

when a step change in boundary displacements or forces is applied at the initial time.

The first order ordinary differential equation solver module is acessed using the macro command ODE1 followed by one of the following macro commands:

INIT to specify the initial displacement vector and the integration constants

LINE to perform the two-time level algorithm

QUAD to perform the three-time level algorithm

Since the matrices [K] and [C] and the vector {F} must be formed before the time integration, the macro commands TANG, CMAS or LMAS and FORM are closely associated with the use of ODE1. It should be noted that the macro FORM forms the vector

$$\{R(T_n)\} = -\{F_n\} - [K_n]\{T_n\}$$

which is allways dependent on the current displacement. Therefore the macro command FORM must be used every time step and preceed ODE1.

To solve a fully nonlinear problem the following macro program may be used:

```
DT          Δt
ODE1 INIT
LOOP        n
TANG
CMAS ( or LMAS )
FORM
ODE1 LINE ( or QUAD )
```

TIME

DISP

NEXT

The macro command program must be followed by the data cards necessary to specify the integration constants θ, γ and β and the initial displacement vector.

If the matrices [K] and/or [C] are not temperature or time dependent then the instructions TANG and/or CMAS or LMAS must be placed outside the loop.

For the simplest case of a linear problem, the macro program may be

```
DT        Δt
ODE1 INIT
TANG
CMAS ( or LMAS )
LOOP      n
FORM
ODE1 LINE ( or QUAD )
TIME
DISP
NEXT
```

V. NUMERICAL EXAMPLES

A. RADIAL TEMPERATURE DISTRIBUTION IN A HOLLOW CYLINDER

Consider the hollow cylinder of infinite length already discussed in section III. Consider the case in which the inside wall is maintained at a constant temperature and heat is lost by convection through the outer wall. This problem was considered for solution by Lew [Ref. 3].

The geometrical and thermophysical characteristics are:

Inside radius (r_{in})	0.1 m
Outside radius (r_{out})	0.3 m
Thermal conductivity (k)	10 W/m °C
Outside ambient temperature	20 °C
Heat transfer coefficient	200 W/m^2°C
Inside wall temperature (T_{in})	820 °C

The finite element model of this problem was completly discussed before and is shown in Figure 1. The complete computer output is given in Appendix B. It must be noted that the element arc width used is arbitrary since the problem is symmetric and no heat is conducted perpendicular to the radius.

The comparision between the analytical and FEAP solutions is shown in Figure 15. In this case the values provided by the finite element model used are very accurate and no further mesh refinement is necessary.

B. NONLINEAR STEADY STATE HEAT CONDUCTION IN A SLAB WITH TEMPERATURE DEPENDENT CONDUCTIVITY

A liquid is boiled by a flat electric heater plate of thickness 2L. The internal energy Q generated electrically may be assumed to be uniform. The boiling temperature of the liquid, corresponding to a specific pressure, is T_W.

In the finite element analysis for the temperature distribuition in the heater, ten equal length quadratic elements were used to represent the unit cross section through the half plate thickness L, since the problem is symmetric. The extremity of the resulting slab is assumed at the temperature T_W while the other is insulated.

The thermal conductivity is assumed temperature dependent according to the expression

$$k = a(1+bT)$$

where a and b are constants.

Assuming the following values for the parameters in a consistent system of units

$T_W = 0.0$

$a = 0.5$

$L = 1.0$

$Q = 1.0$

the problem was solved for b equal 0.0, 1.0 and 2.0.

The numerical results are compared with the analytical solutions in Figure 16 and one may conclude that FEAP provides exact solutions in this problem.

C. TRANSIENT TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

The same hollow cylinder of infinite length treated before is considered again but the case solved is the one with both outside and inside walls maintained at constant temperatures. Thermal diffusivity, $\alpha = k/\rho c$ is specified as 0.00005 m^2/sec. Initially the cylinder is at an ambient temperature T_0 of 20°C; suddenly the temperature of the inside wall is raised to 820°C while the outside wall is maintained at 20°C. All other conditions are the same as for the steady state problem.

In order to obtain the unsteady solution the number of elements in the finite element model was doubled, thus twelve radial quadratic elements were used. The element arc width chosen is the same 6° as before.

The two time-level scheme with $\theta = 2/3$ was used for the time integration. In order to compare the results the dimensionless temperature T*, radius r* and Fourier number Fo were used. They are defined as

$$T^* = (T - T_0)/(T_{in} - T_0)$$

$$r^* = r/r_{in}$$

and

$$Fo = \alpha t/r_{in}^2$$

A time step size Δt of 2 seconds was chosen initially. After 10 sec, Δt was increased to 10 sec and to 50 sec after 200sec. After 1000 sec, Δt was increased to 100 sec.

Figure 17 shows the evolution of the temperature of the

center interior points of the cylinder and Figure 18 compares the analytical values with the FEAP solution for the radial temperature distribuition for two different times.

D. SLAB WITH RADIATION-CONVECTION BOUNDARY CONDITIONS

Consider a plane slab of thickness L, with constant thermophysical properties ($k= \rho c=1$), subjected to simultaneous radiation and convection at $x=0$ and insulated at $x=L$ ($0<x<L$). The slab is initially at a uniform temperature T_0 and it was decided to consider zero ambient temperatures for convection and radiation.

It was also decided to consider the Biot number

$$Bi = h_c L/k = 1$$

where h_c is the convection heat tranfer coefficient and

$$R = \varepsilon\sigma T_0^3 L/k = 4$$

where ε is the emissivity and σ the Stefan-Boltzman constant.

The same mesh as in Example B was employed for the finite element solution. The time integration was performed by the three-time level scheme with $\gamma=1/2$ and $\beta=1/3$. A variable time step size was chosen. A value of $\Delta t=0.001$ was used initially. After $t=0.02$, Δt was increased to 0.0025. After $t=0.2$, Δt was increased to 0.01. After $t=0.4$, Δt was increased to 0.025. After $t=1.0$, Δt was increased to 0.05.

The history of the ratios T_w/T_0 and T_m/T_0 are plotted in Figure 19 against the Fourier number

$$Fo = \alpha t/L^2$$

T_w is the surface tmperature at $x=0$ and T_m is the surface temperature at $x=L$.

The finite element results are compared with those given by Haji-Sheick and Sparrow in Ref. 12 and obtained from a Monte Carlo method.

VI. CONCLUSIONS AND RECOMMENDATIONS

The computer program in this thesis provides an accurate and reliable means for solving a variety of Heat Transfer problems. The use of this program and efforts to increase its versatility are highly encouraged.

The capabilities of the program, while quite significant, can still be improved. The present version is designed for an "in-core" solution technique, which restricts the problem size to within the computer core-size. The capacity to handle large problems may be increased through the use of some "out of core" technique for solving the system of equations and even for building the mesh data. In that case the size of the problems treated would be restricted only by the availability of external storage devices.

In order to check the mesh input and to easily interpret the output, a graphics module must be included in the program.

Although the time integration module seems to provide reliable results in linear and some nonlinear problems, it should be subjected to further research in order to test it with different types of nonlinear analysis.

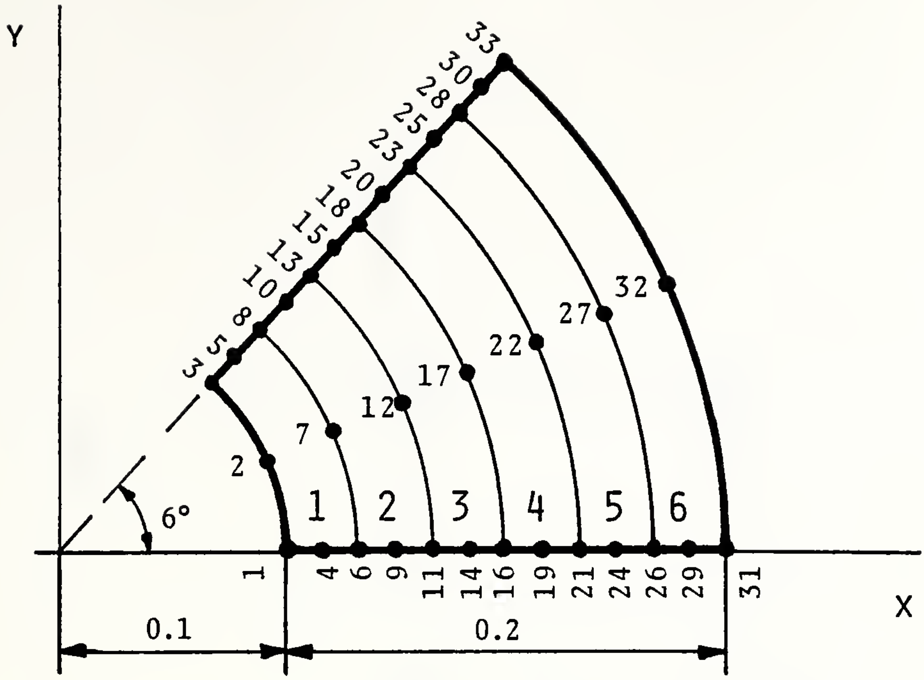


Figure 1. Mesh for Hollow Cylinder Problem

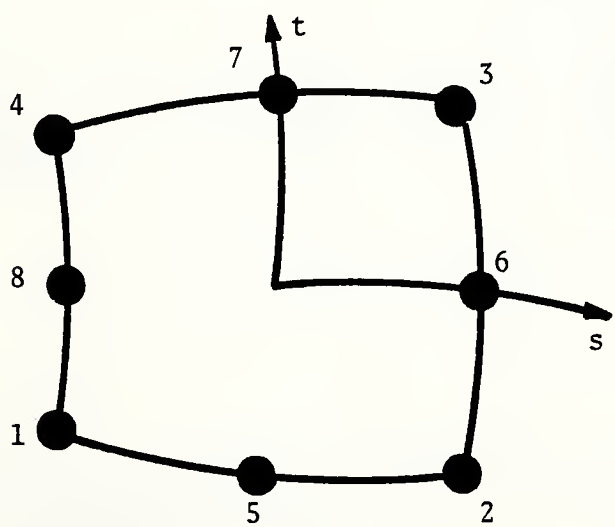


Figure 2. Eight-Node Serendipity Quadrilateral

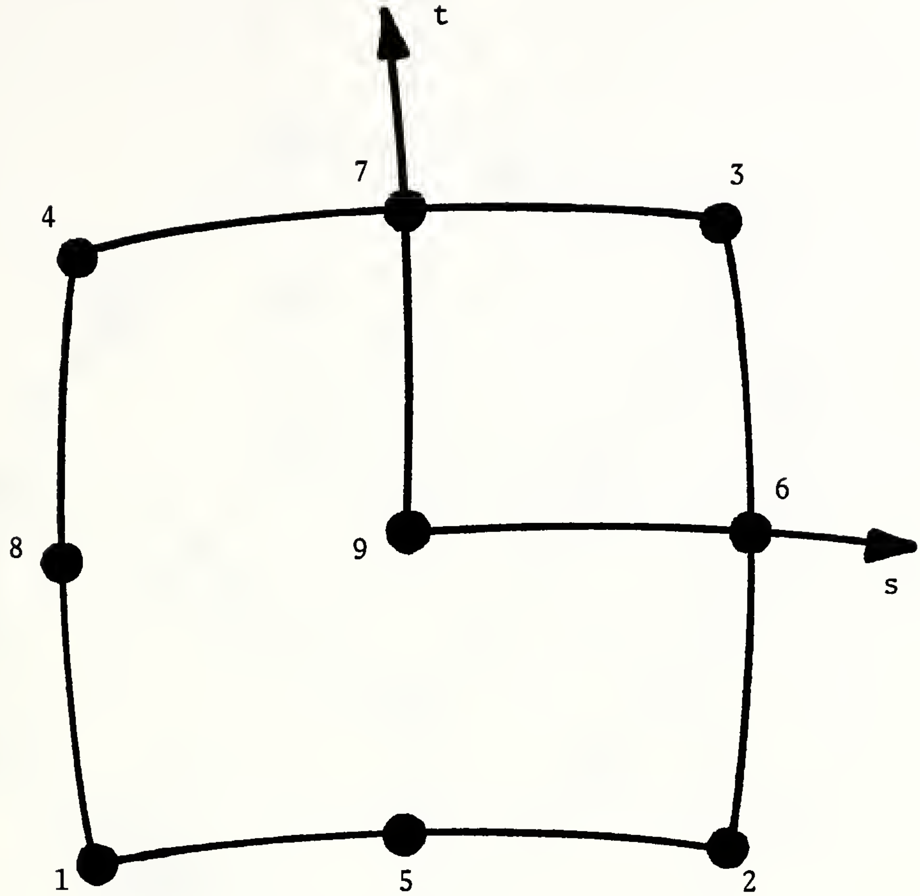


Figure 3. Local Node Number Sequence for a Quadratic
Lagrangian Element

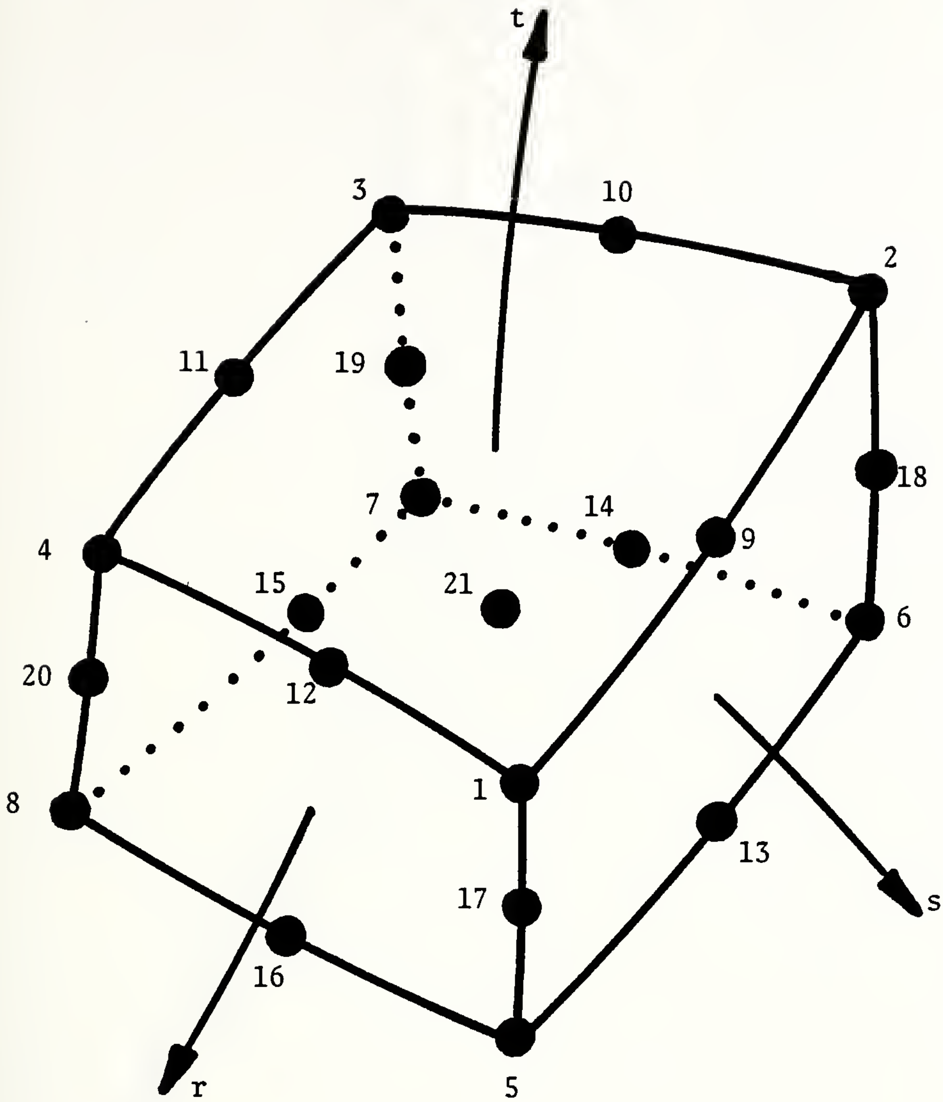


Figure 4. Local Node Number Sequence for a 21-Node Three Dimensional Element

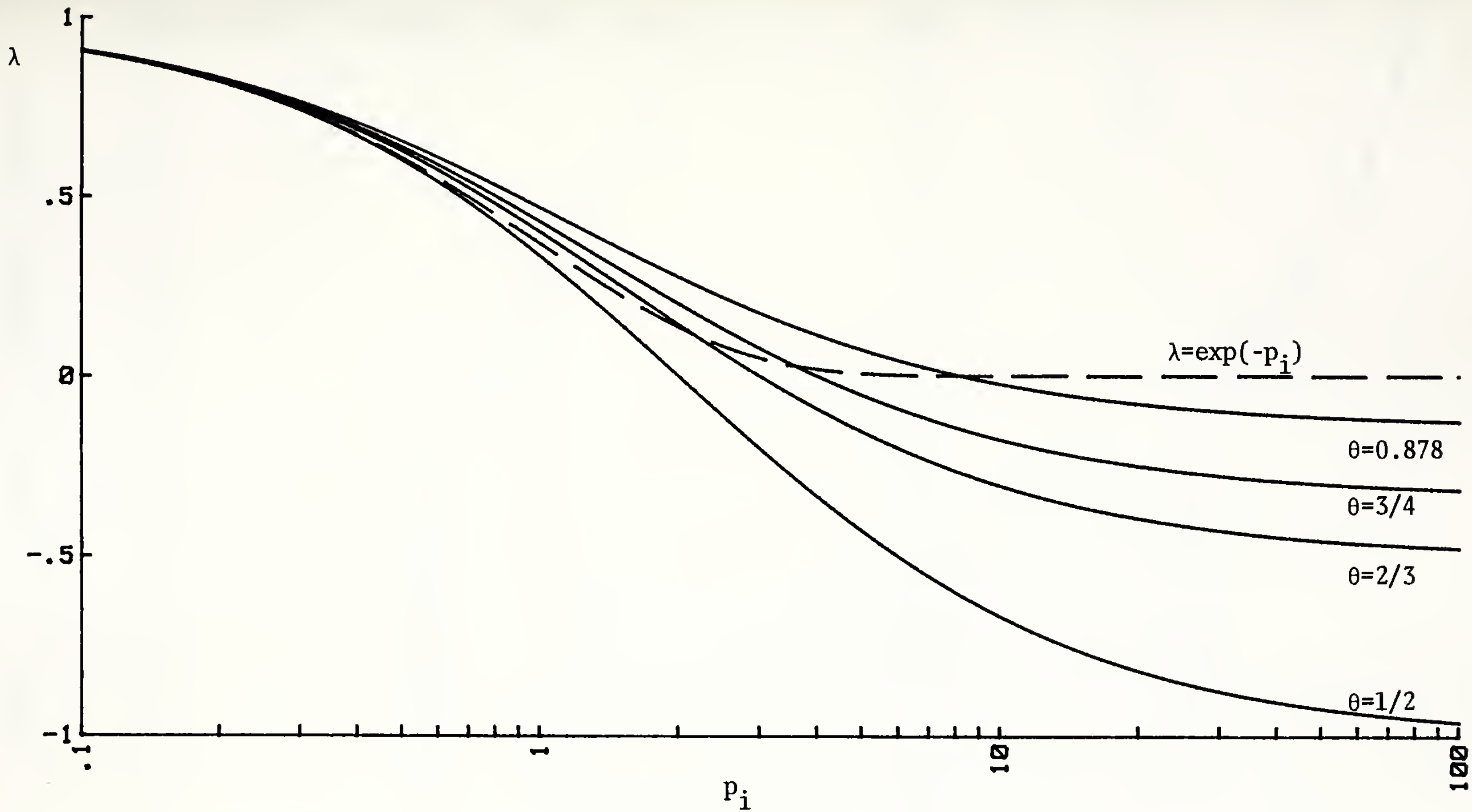


Figure 5. Zienkiewicz Two-Time Level Scheme

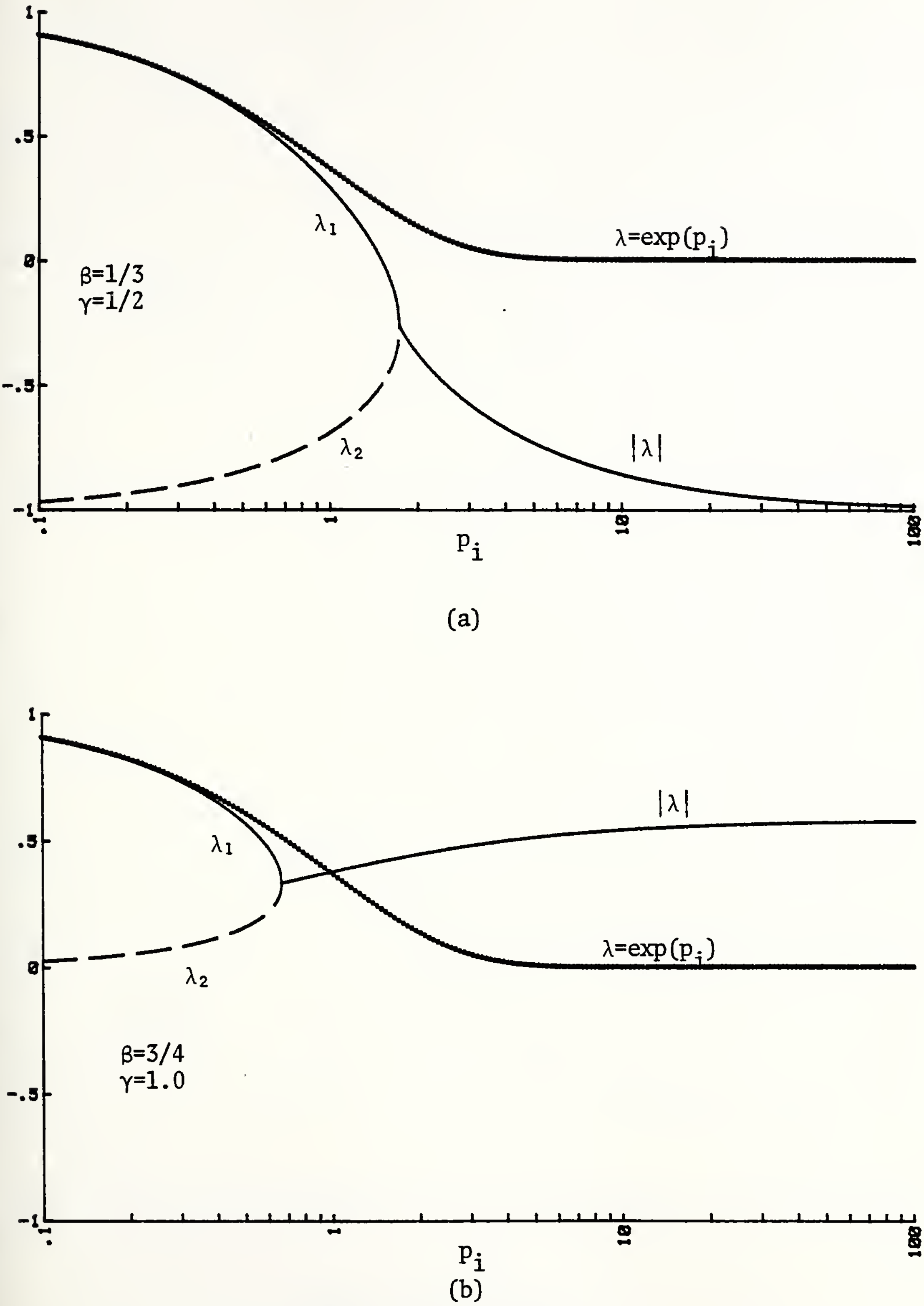
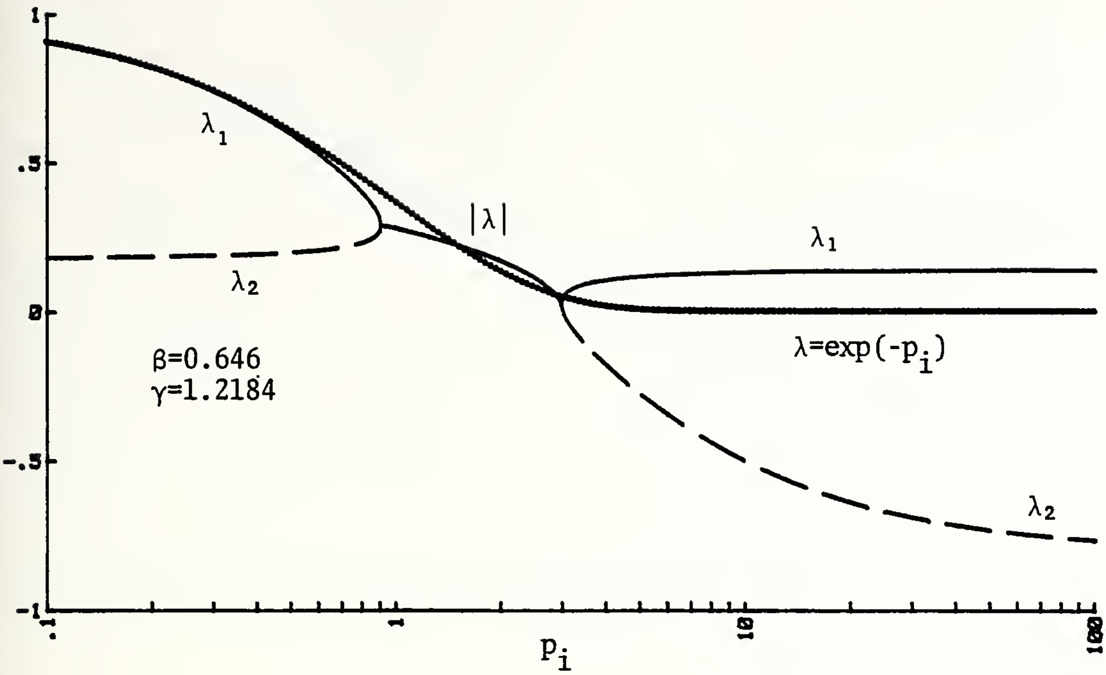
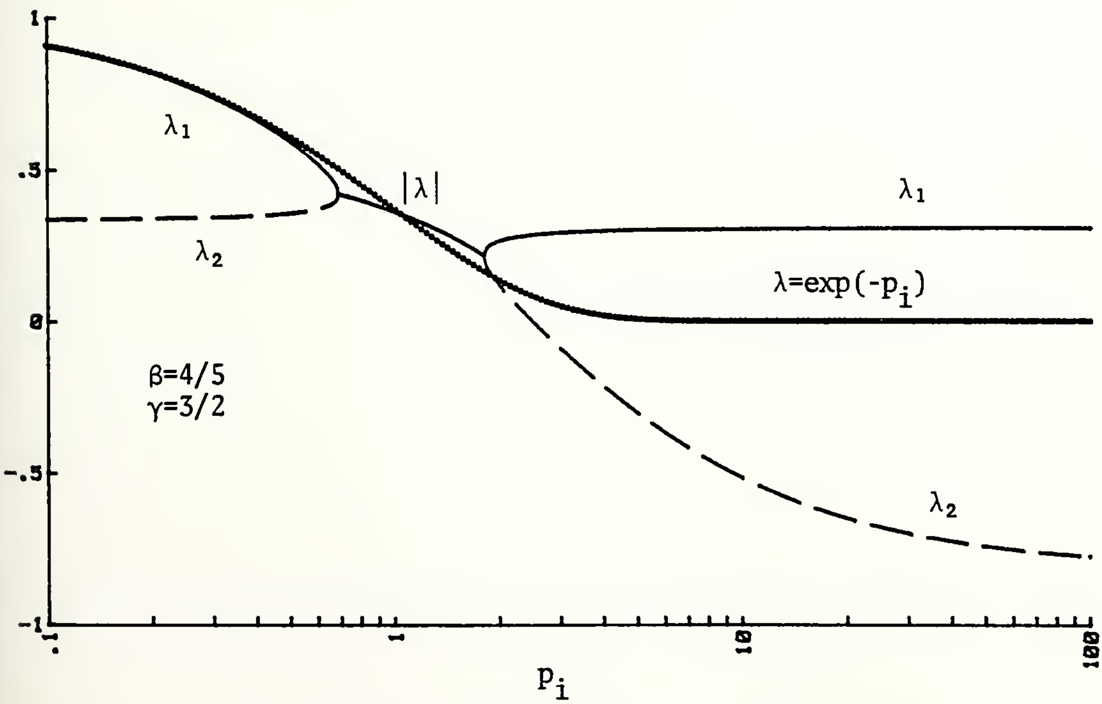


Figure 6. Zienkiewicz Three-Time Level Schemes

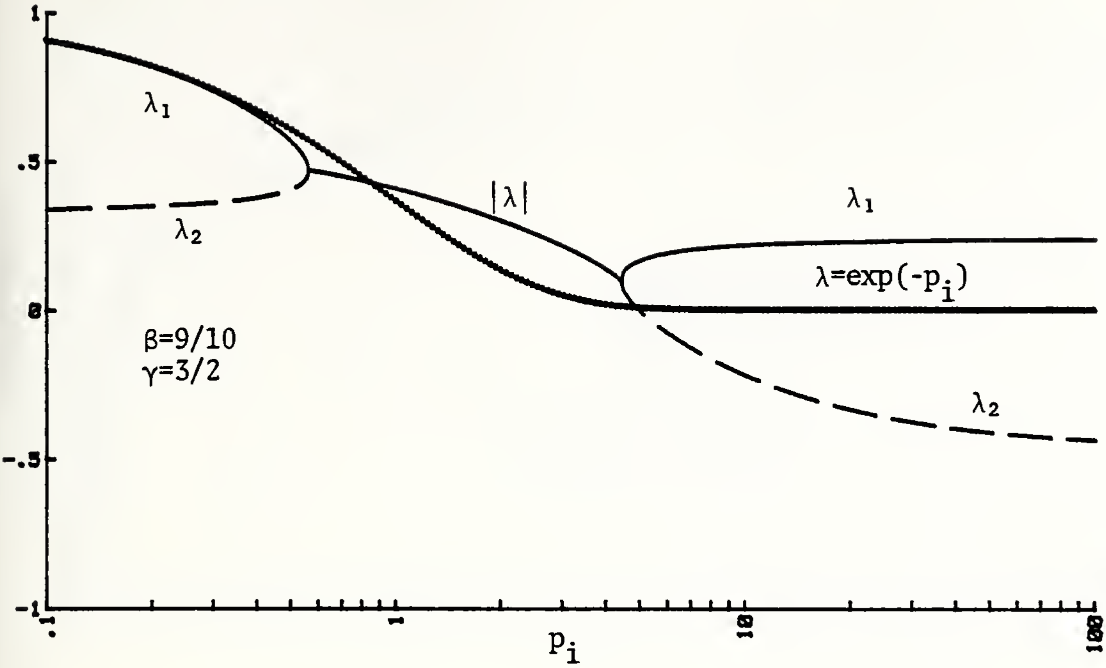


(c)

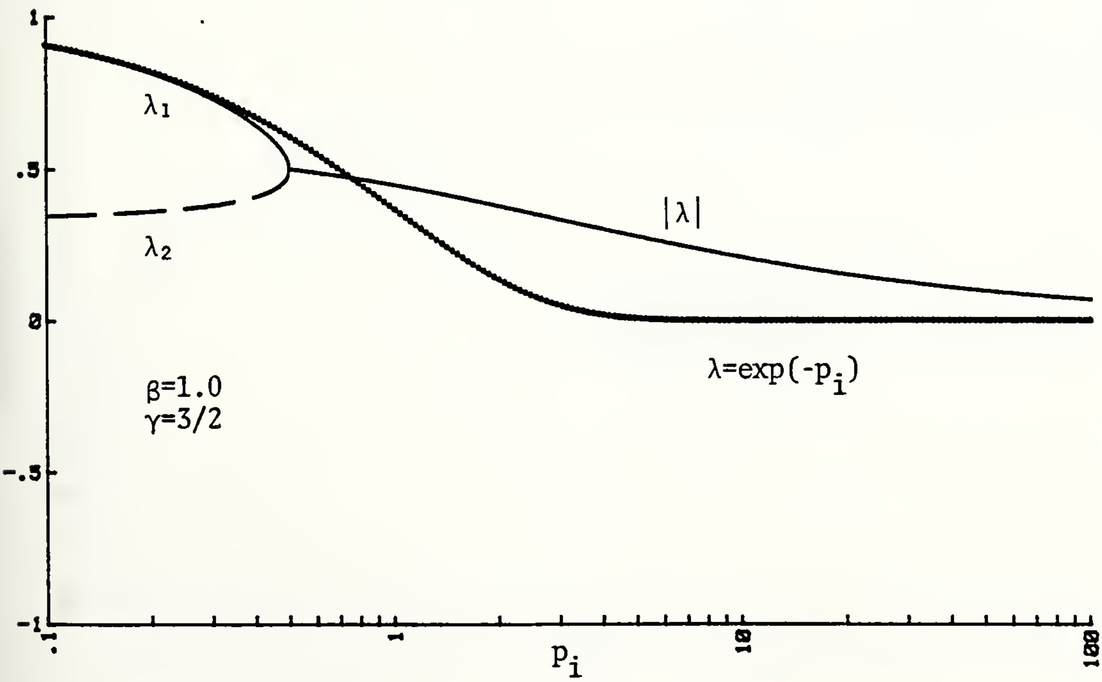


(d)

Figure 6. (continued)

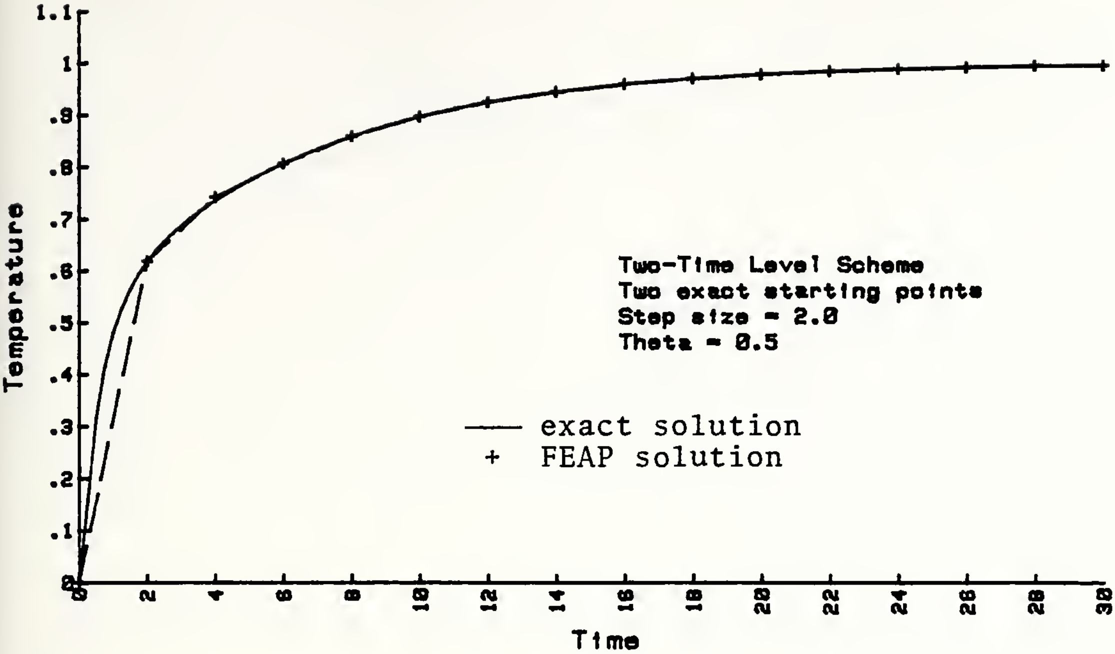


(e)



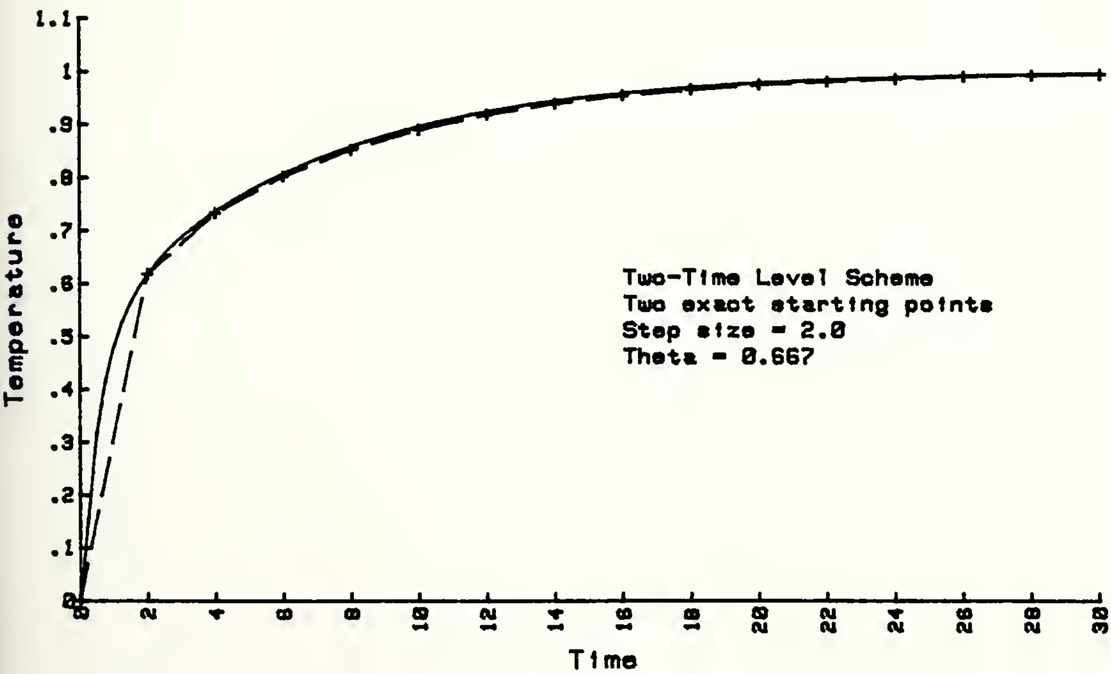
(f)

Figure 6. (continued)



1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Time
Two-Time Level Scheme
Two exact starting points
Step size = 2.0
Theta = 0.5
exact solution
FEAP solution

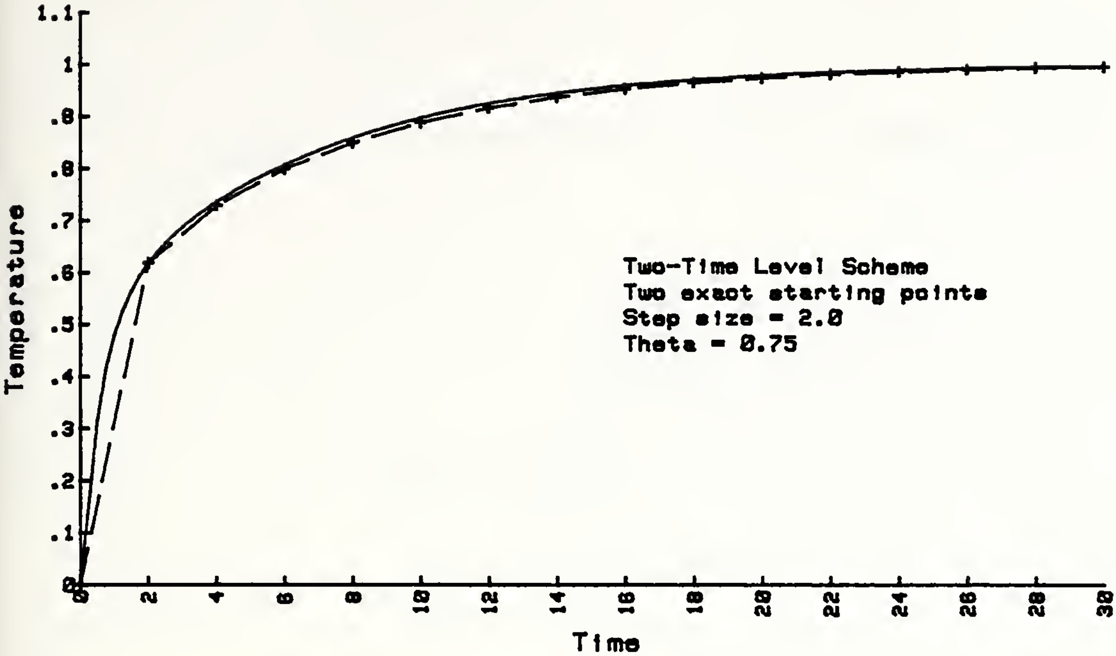
(a)



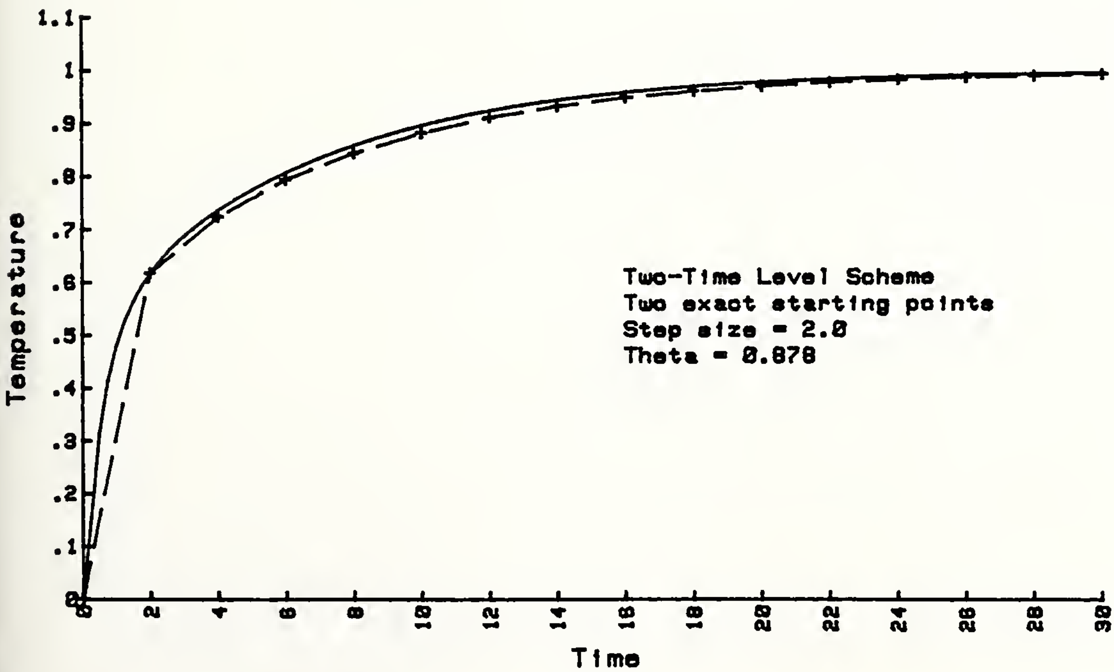
1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Time
Two-Time Level Scheme
Two exact starting points
Step size = 2.0
Theta = 0.667

(b)

Figure 7.

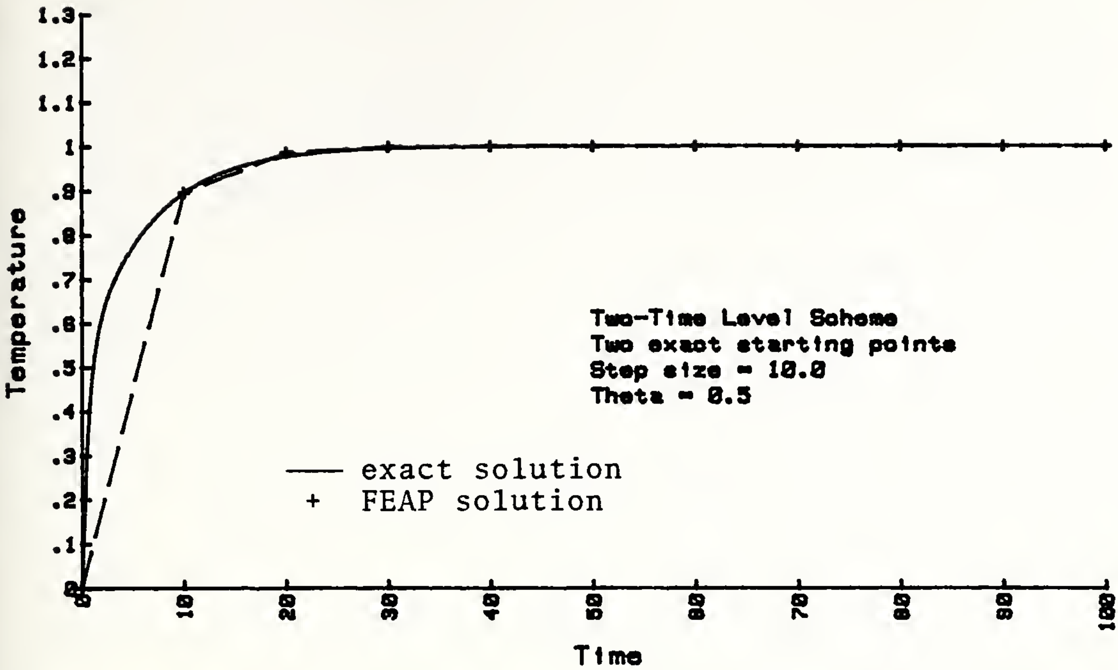


(c)

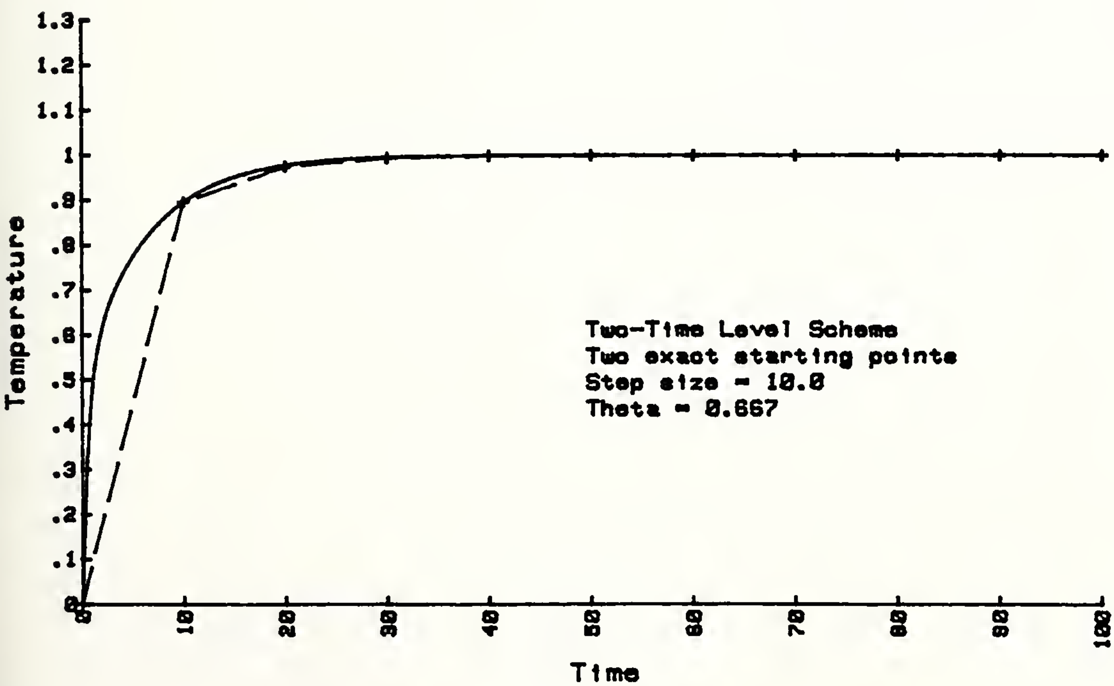


(d)

Figure 7. (continued)

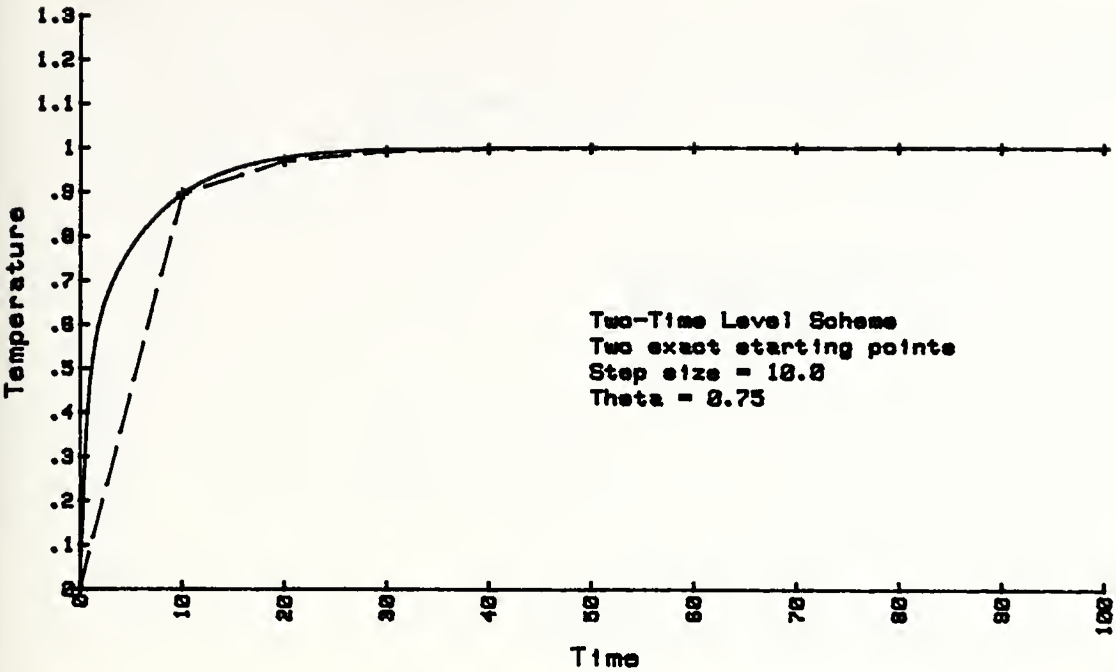

1.3
1.2
1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
0
10
20
30
40
50
60
70
80
90
100
Time
Two-Time Level Scheme
Two exact starting points
Step size = 10.0
Theta = 0.5
exact solution
+ FEAP solution

(a)

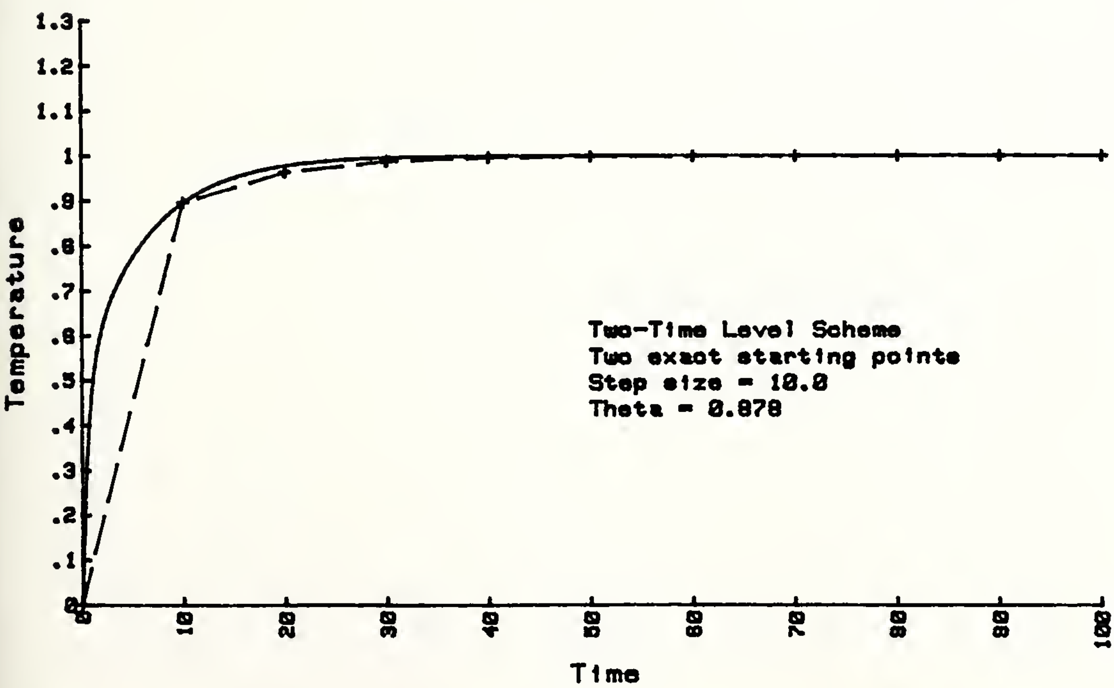

1.3
1.2
1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
0
10
20
30
40
50
60
70
80
90
100
Time
Two-Time Level Scheme
Two exact starting points
Step size = 10.0
Theta = 0.667

(b)

Figure 8.

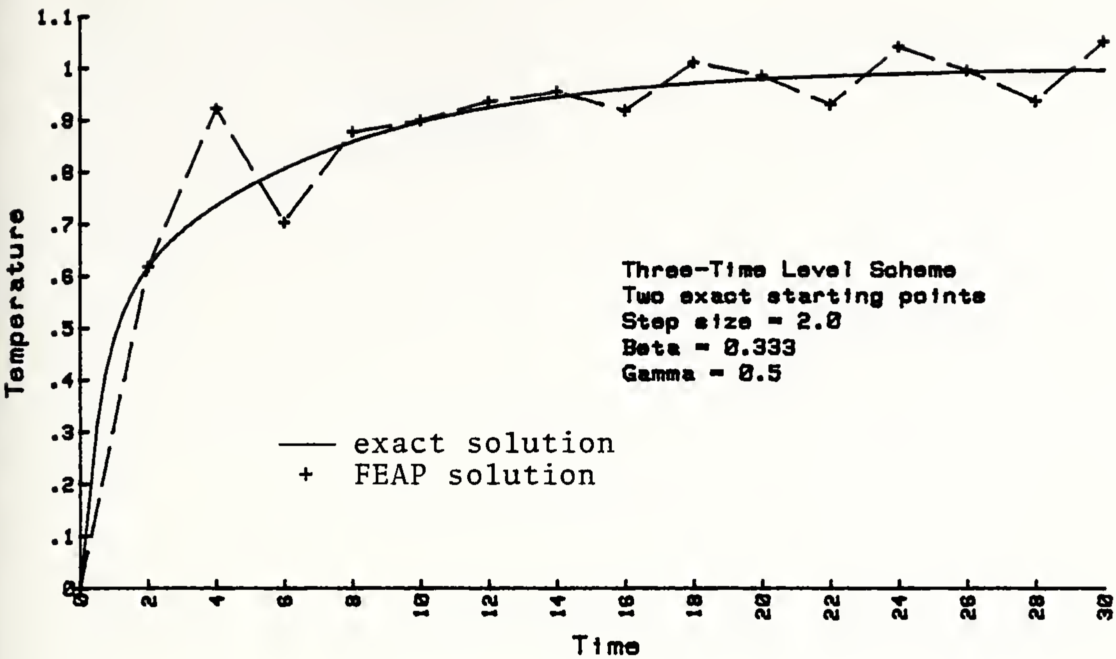


(c)

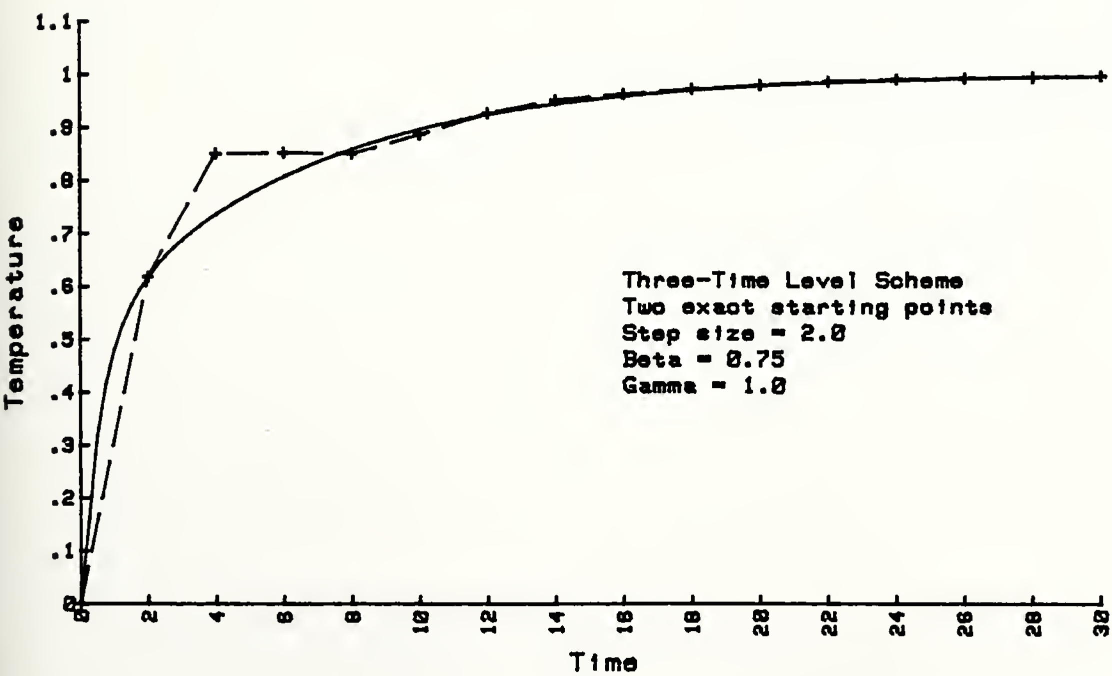


(d)

Figure 8. (continued)

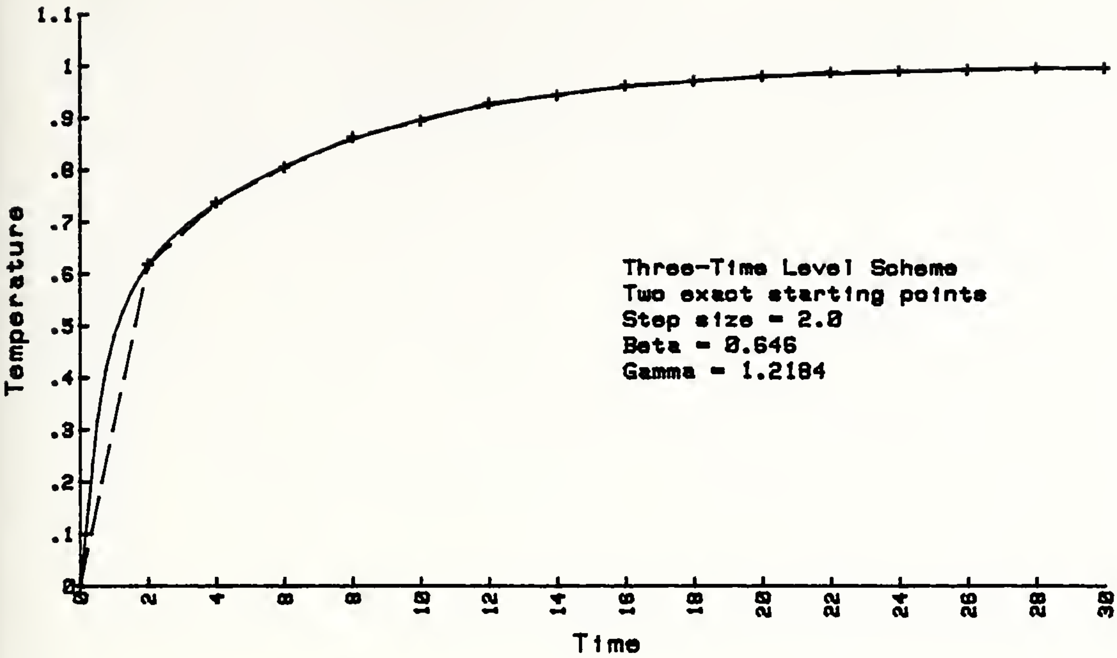

1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Time
Three-Time Level Scheme
Two exact starting points
Step size = 2.0
Beta = 0.333
Gamma = 0.5
exact solution
+ FEAP solution

(a)

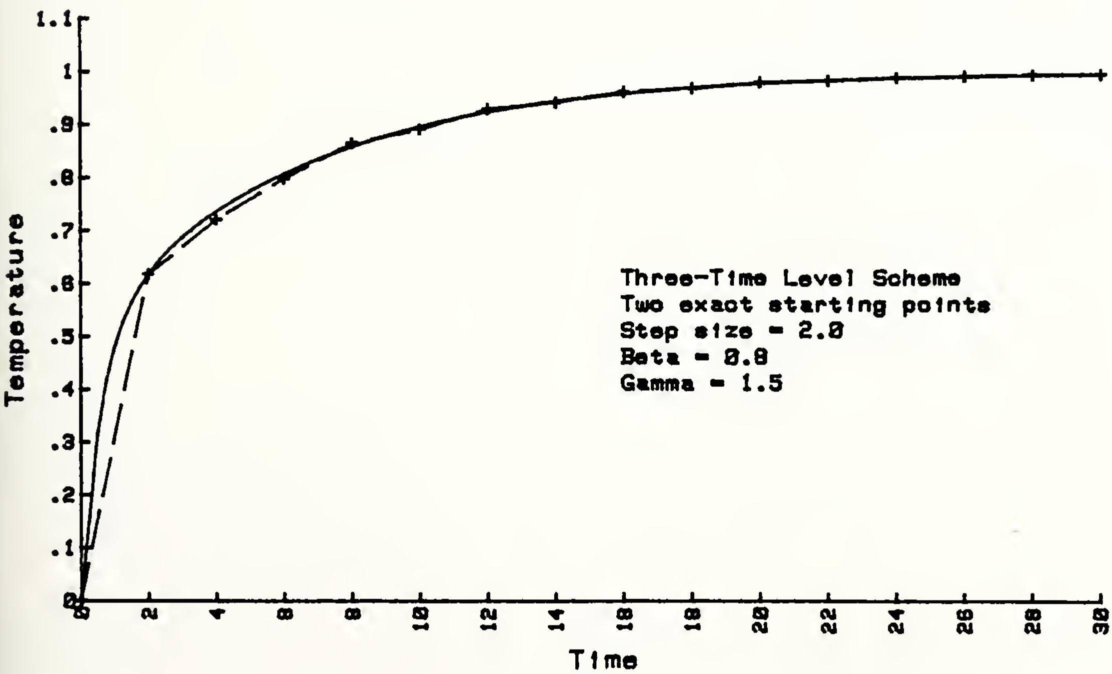

1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Time
Three-Time Level Scheme
Two exact starting points
Step size = 2.0
Beta = 0.75
Gamma = 1.0

(b)

Figure 9.

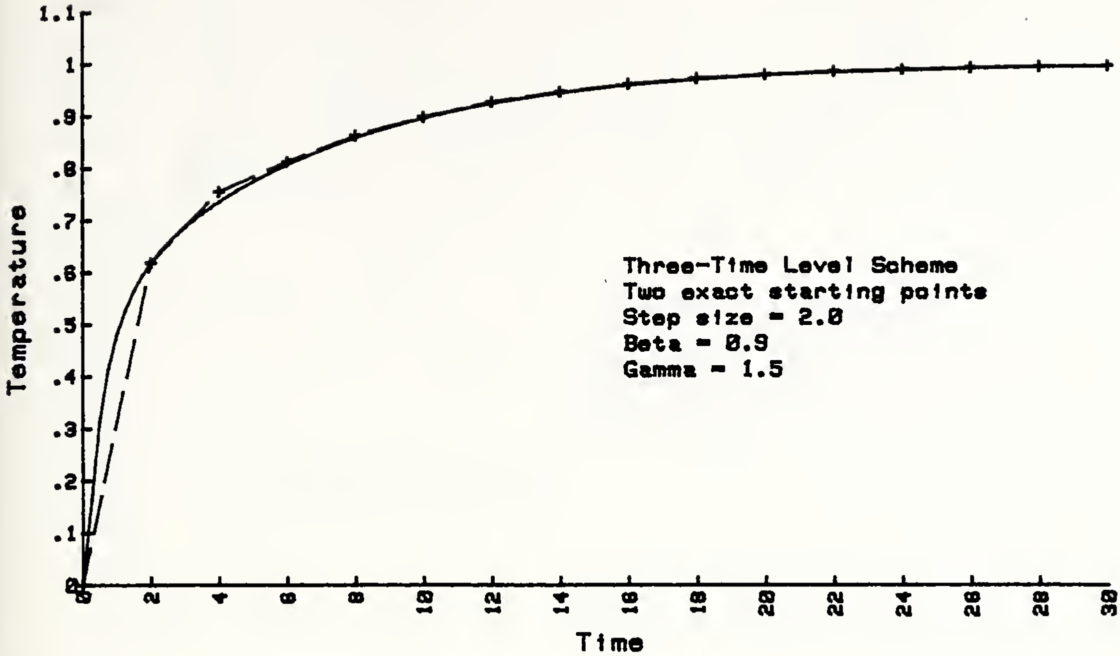


(c)

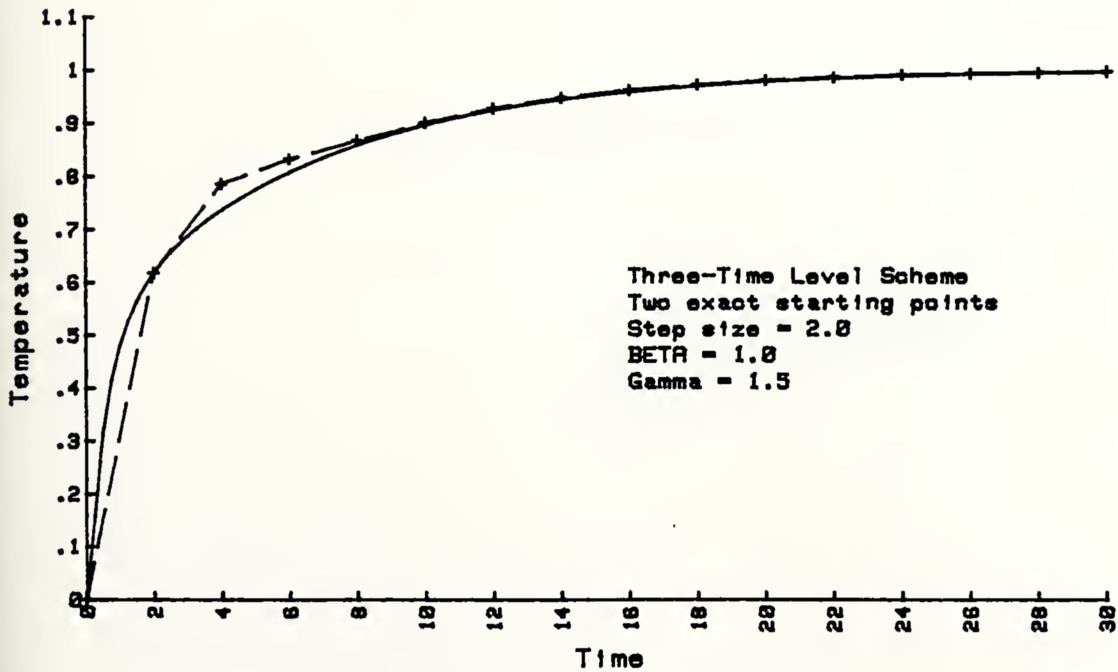


(d)

Figure 9. (continued)

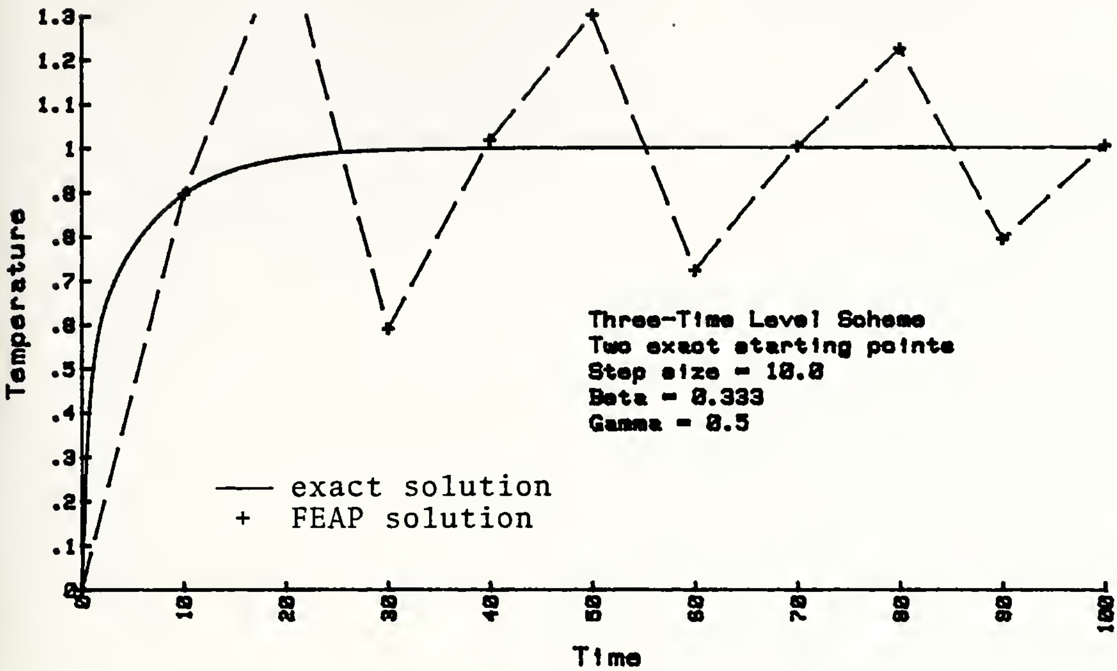


(e)



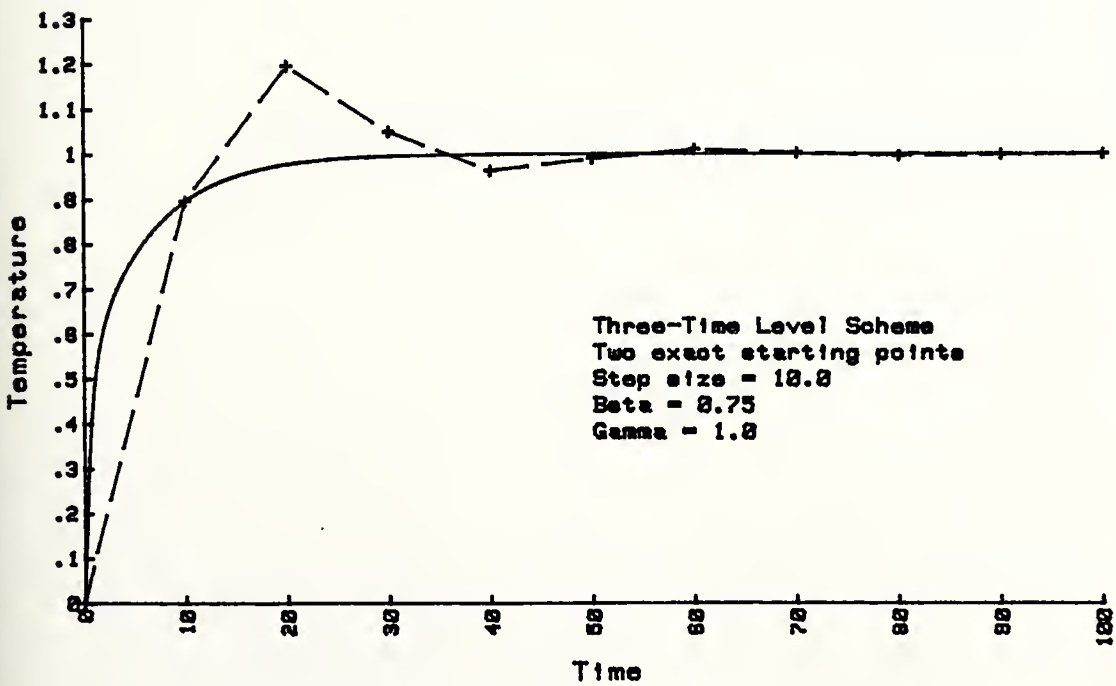
(f)

Figure 9. (continued)



Three-Time Level Scheme
Two exact starting points
Step size = 10.0
Beta = 0.333
Gamma = 0.5
exact solution
+ FEAP solution
Temperature
Time

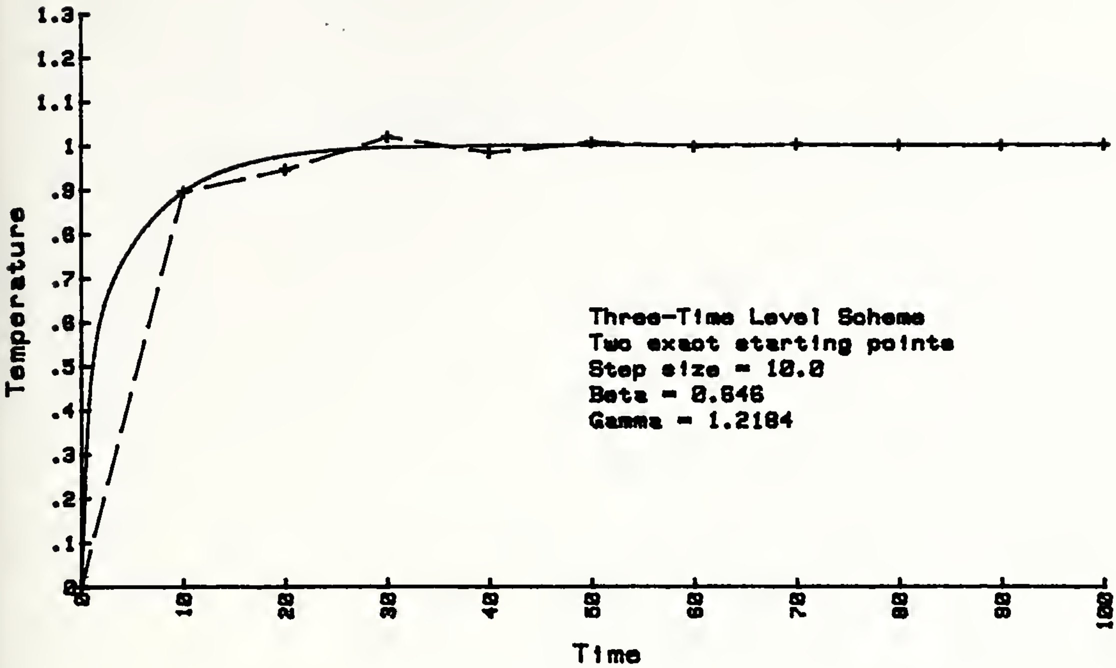
(a)



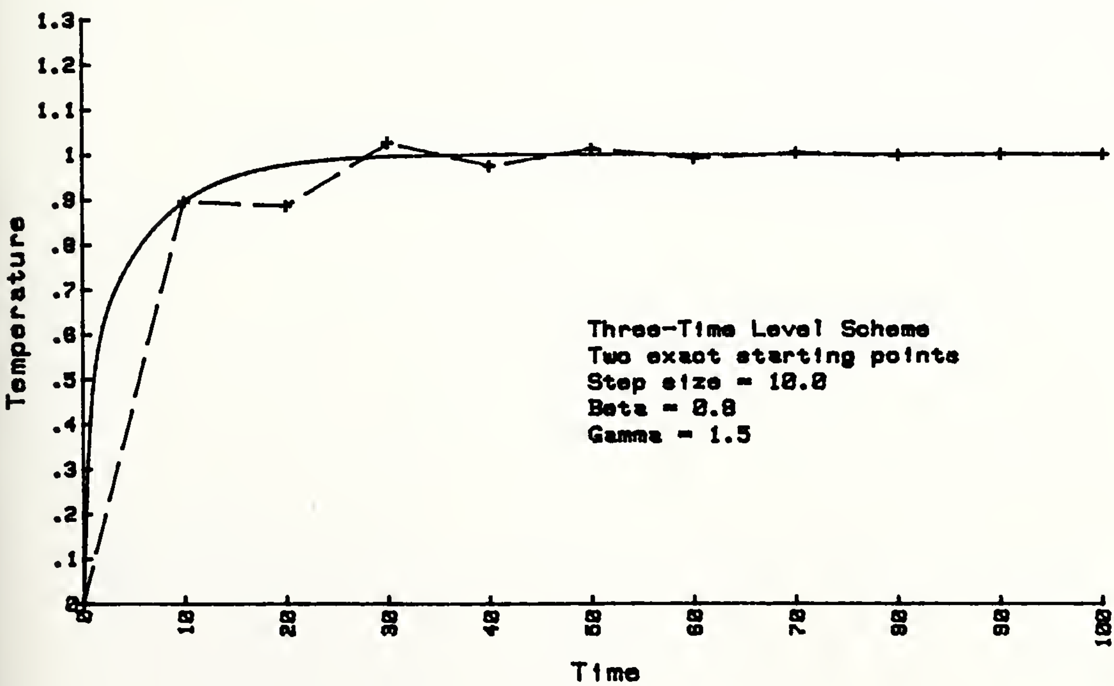
Three-Time Level Scheme
Two exact starting points
Step size = 10.0
Beta = 0.75
Gamma = 1.0
Temperature
Time

(b)

Figure 10.

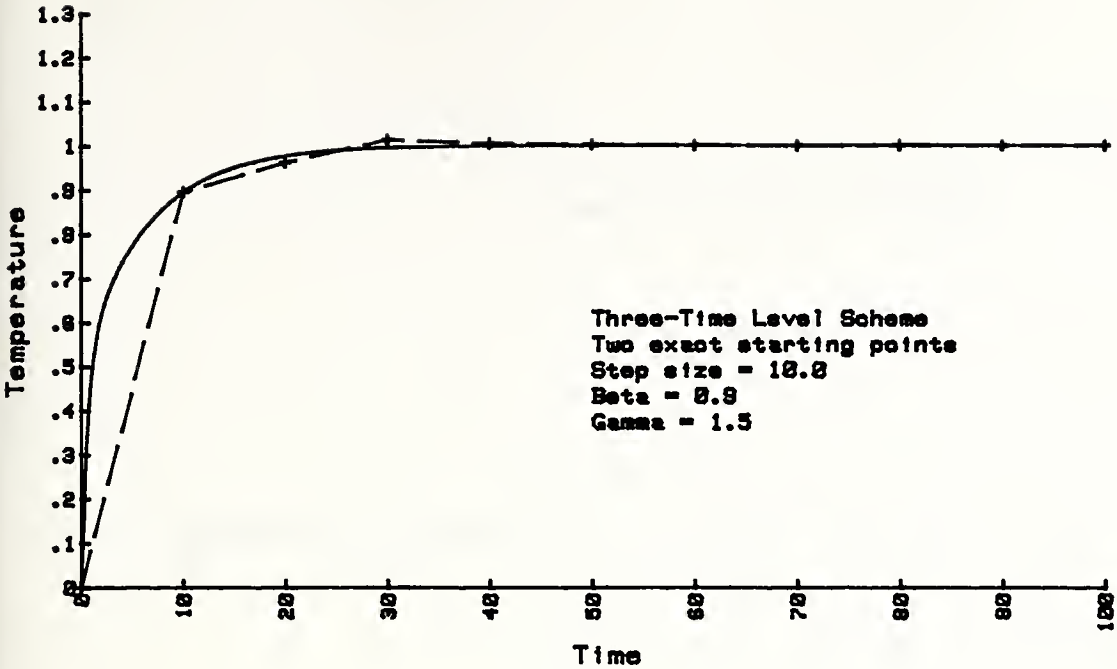


(c)

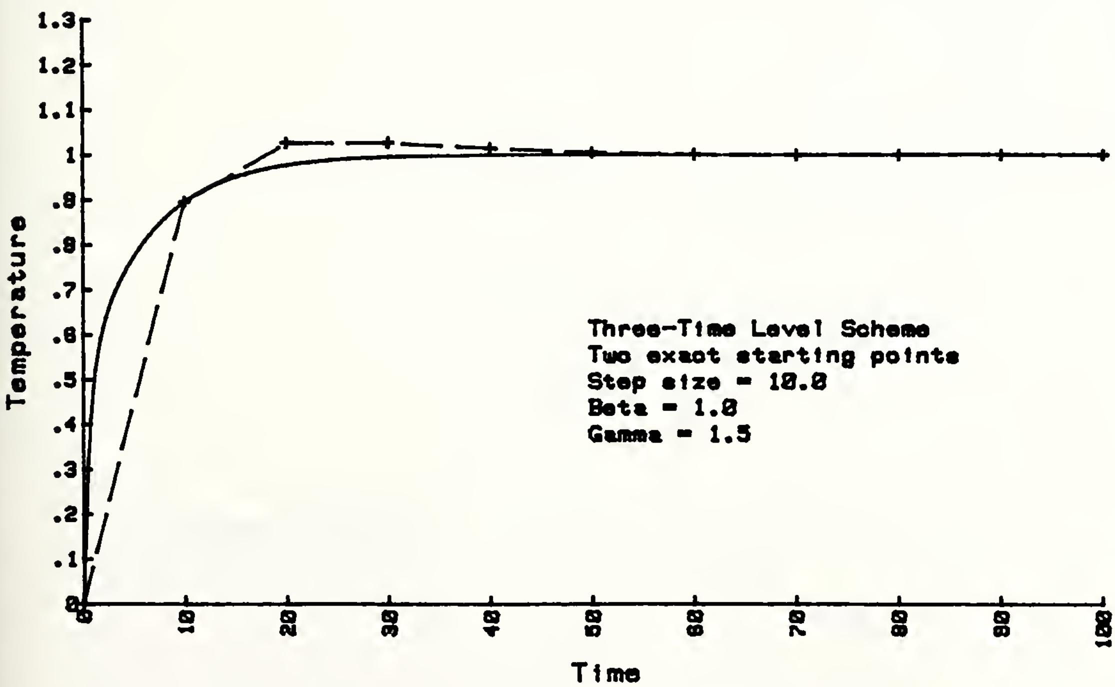


(d)

Figure 10. (continued)



(e)



(f)

Figure 10. (continued)

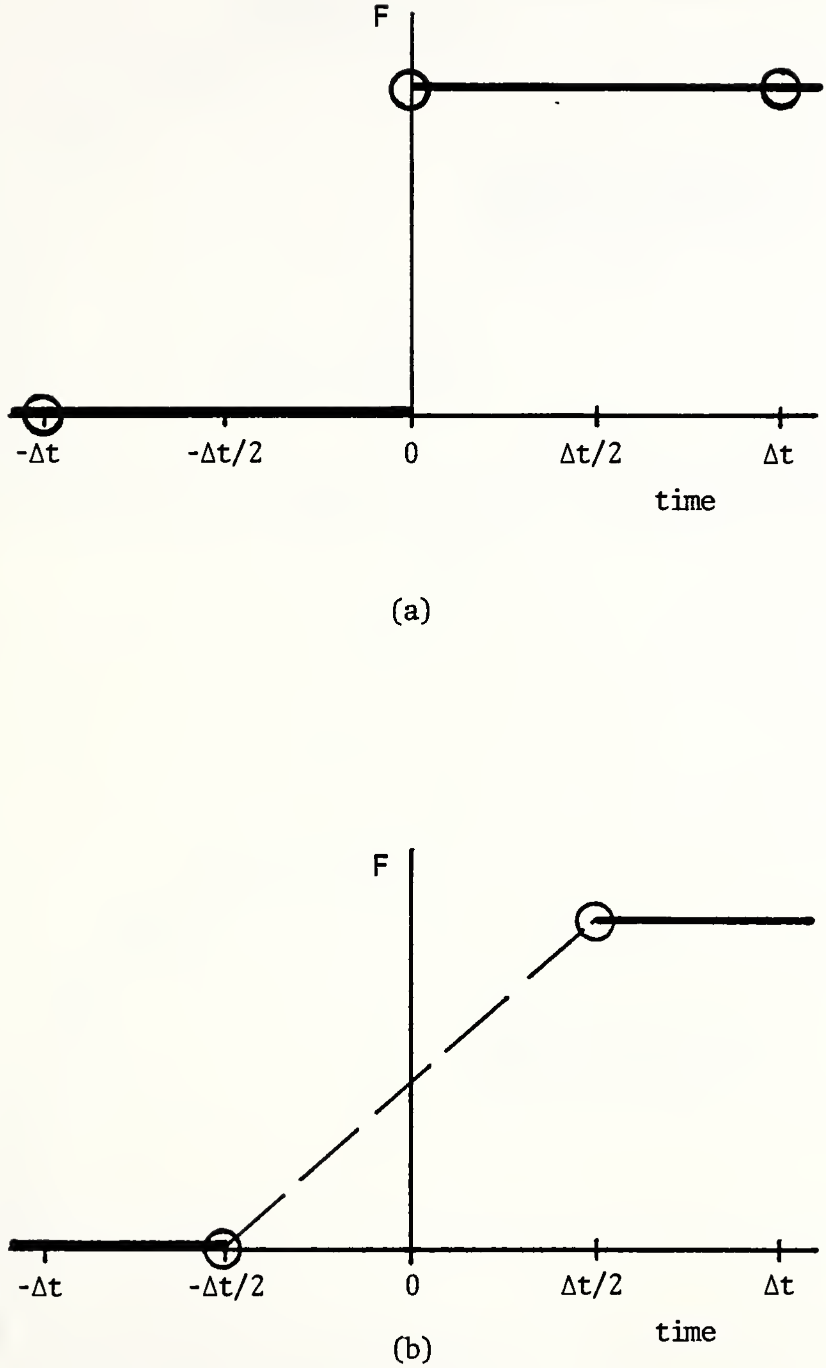
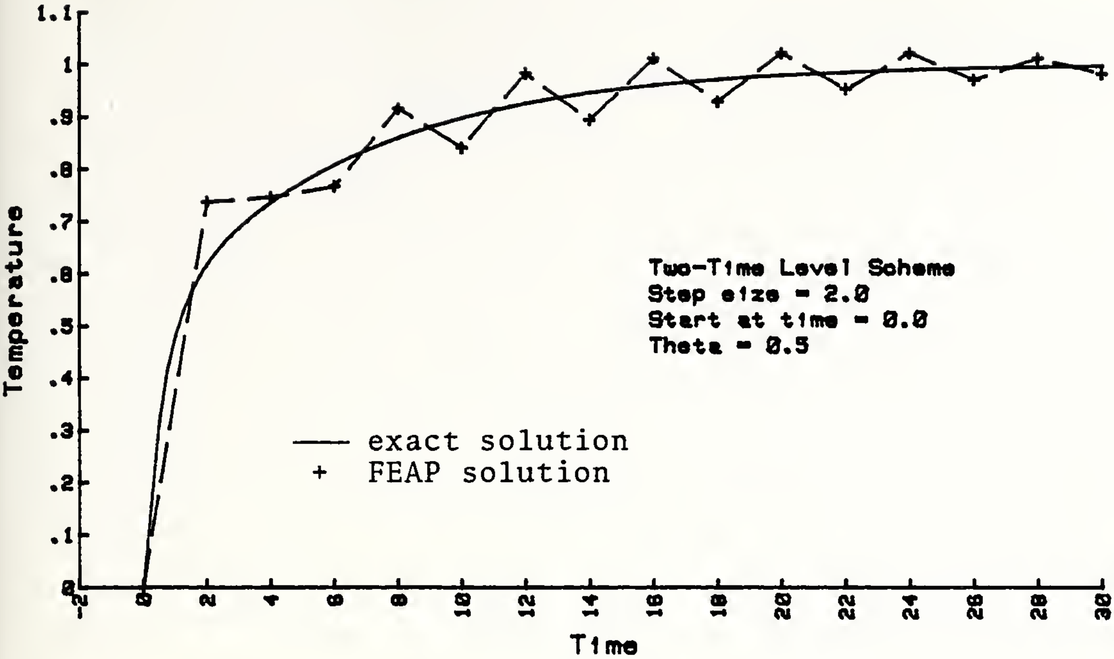
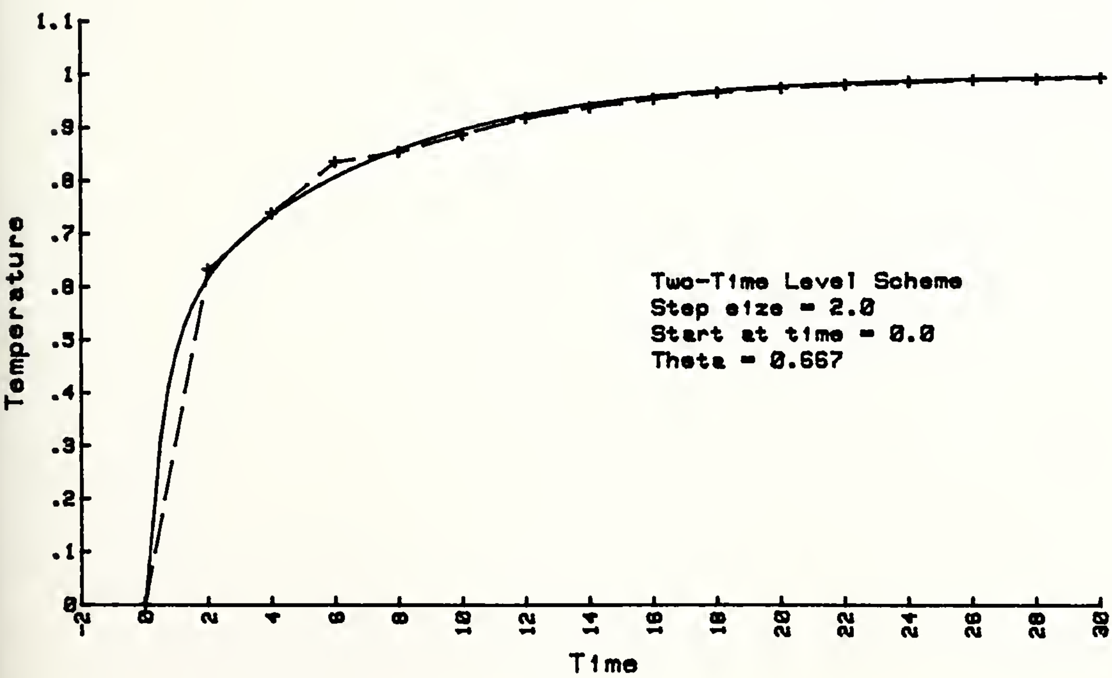


Figure 11. Step Change in the Loading Term

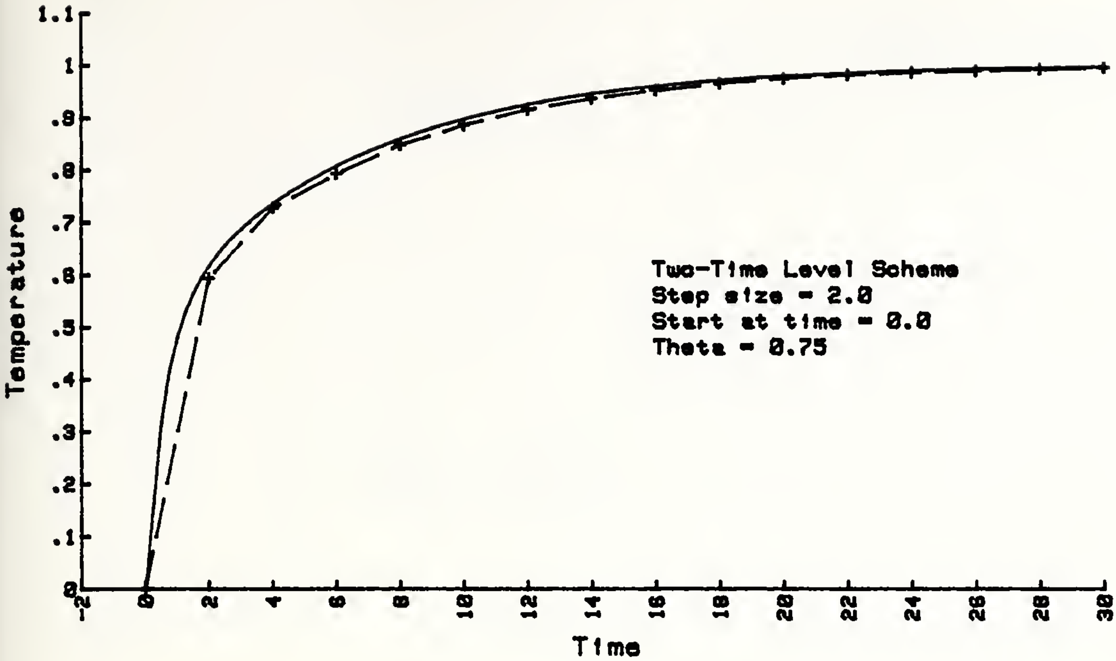

1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
-2
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Time
Two-Time Level Scheme
Step size = 2.0
Start at time = 0.0
Theta = 0.5
exact solution
+ FEAP solution

(a)

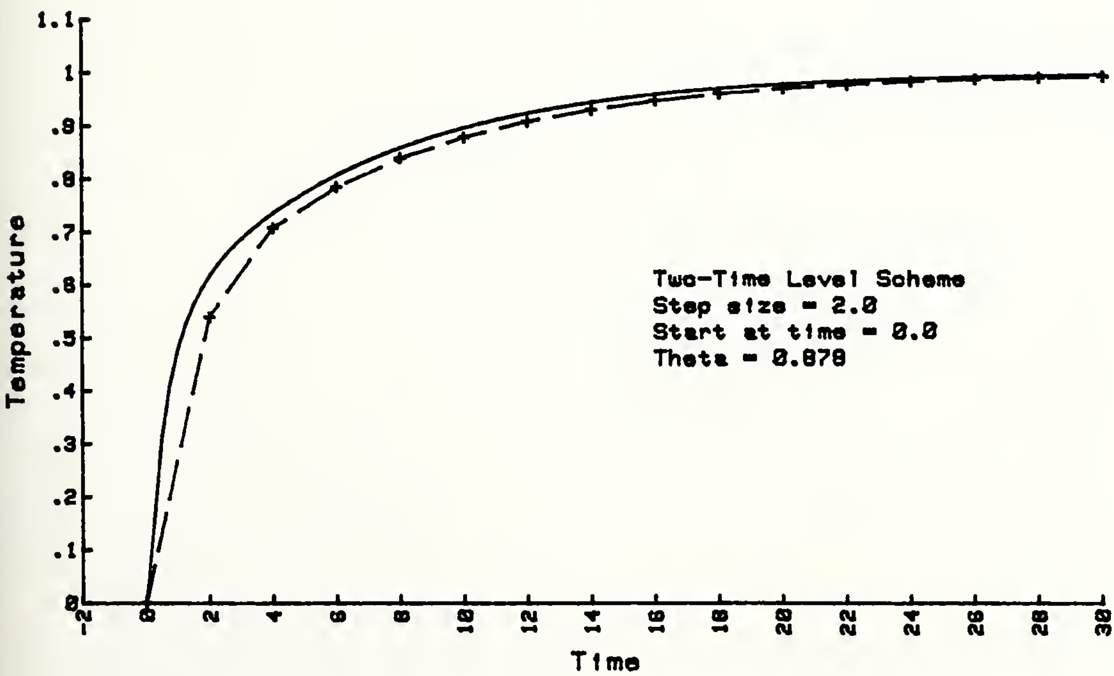

1.1
1
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Temperature
-2
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
Time
Two-Time Level Scheme
Step size = 2.0
Start at time = 0.0
Theta = 0.667

(b)

Figure 12.

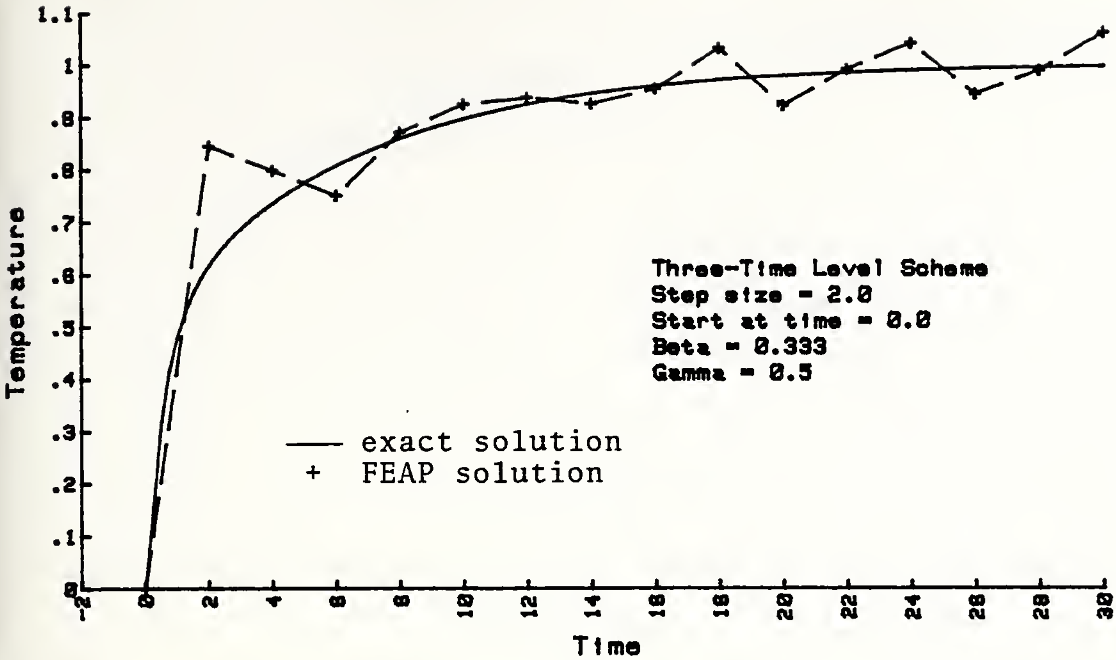


(c)

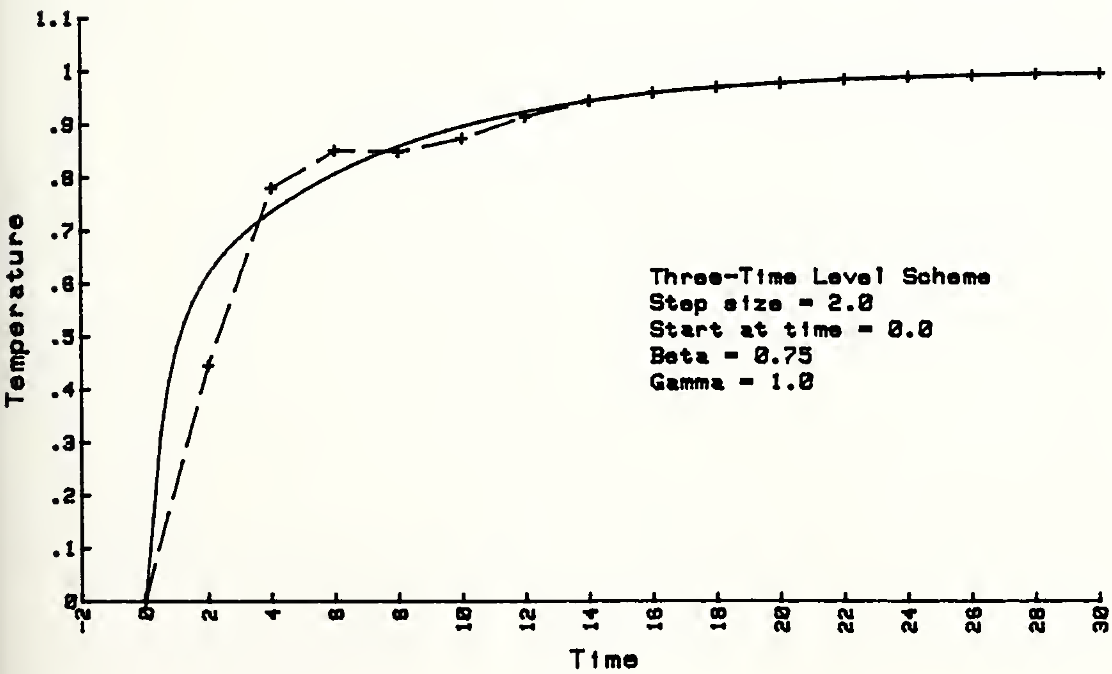


(d)

Figure 12. (continued)

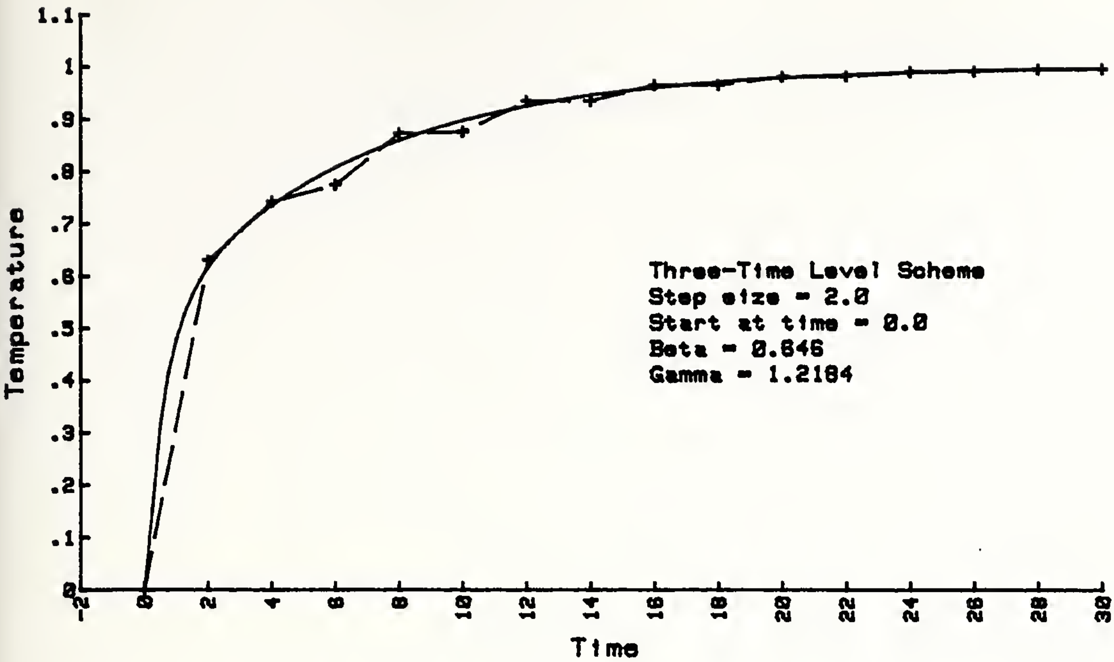

Three-Time Level Scheme
Step size = 2.0
Start at time = 0.0
Beta = 0.333
Gamma = 0.5
exact solution
+ FEAP solution
Temperature
Time

(a)

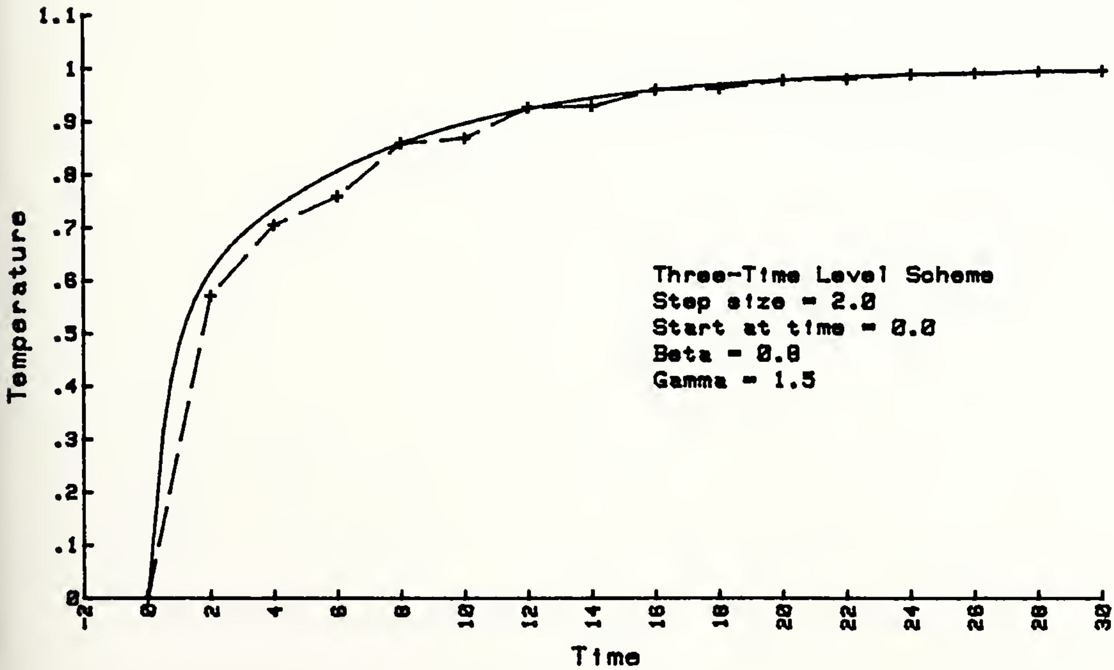

Three-Time Level Scheme
Step size = 2.0
Start at time = 0.0
Beta = 0.75
Gamma = 1.0
Temperature
Time

(b)

Figure 13.

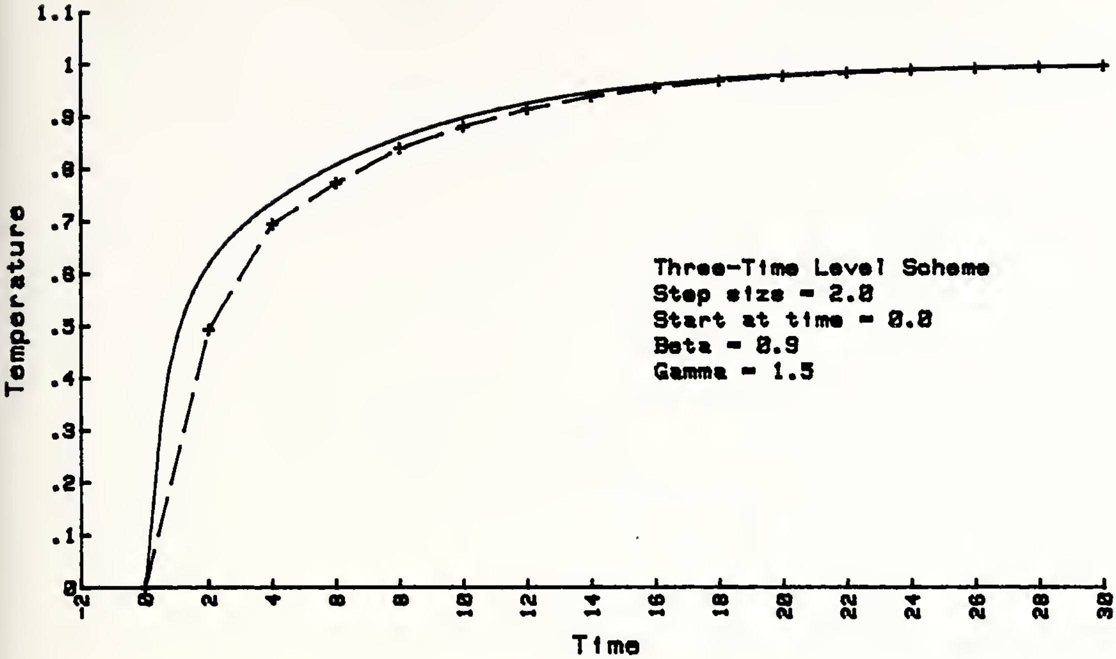


(c)

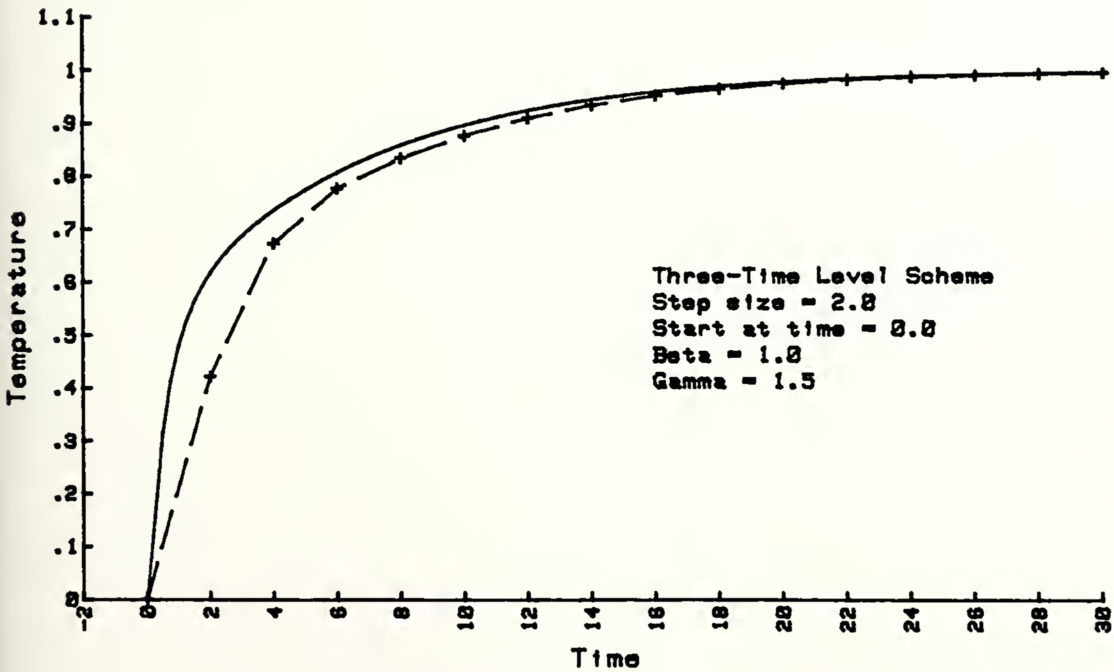


(d)

Figure 13. (continued)

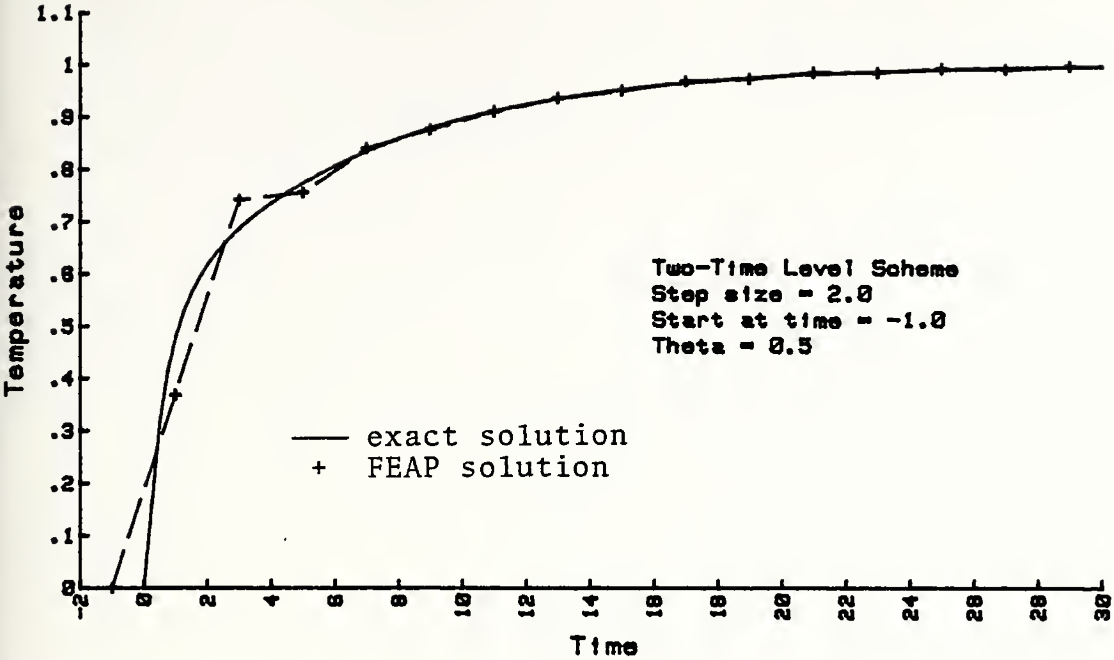


(e)



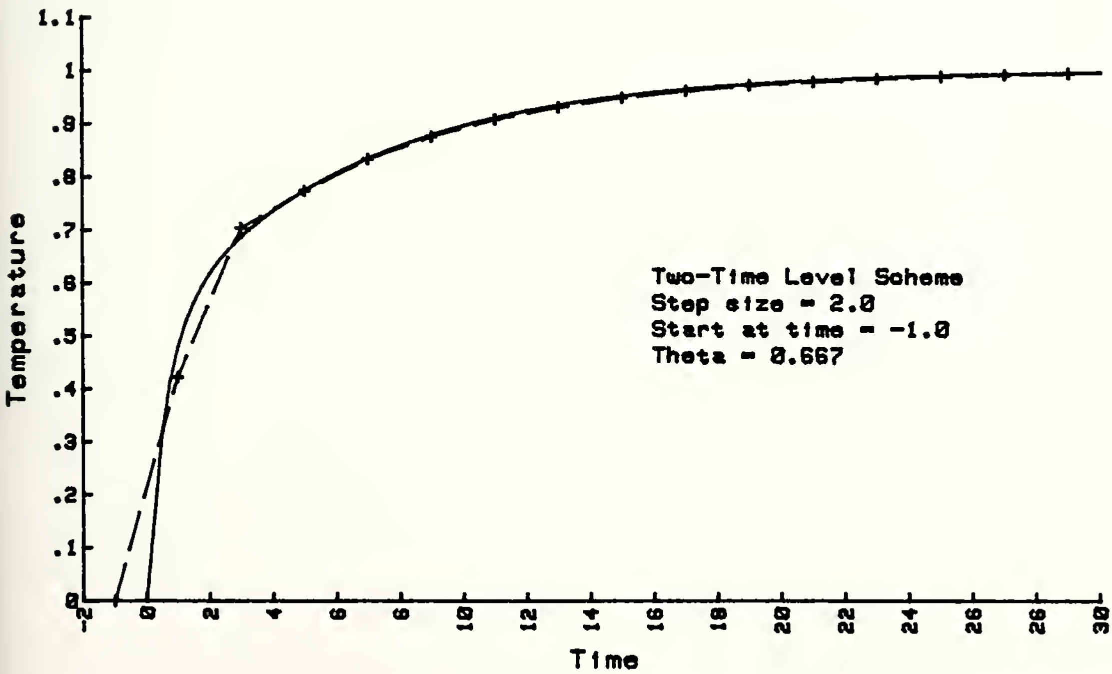
(f)

Figure 13. (continued)



Two-Time Level Scheme
Step size = 2.0
Start at time = -1.0
Theta = 0.5
exact solution
+ FEAP solution
Temperature
Time

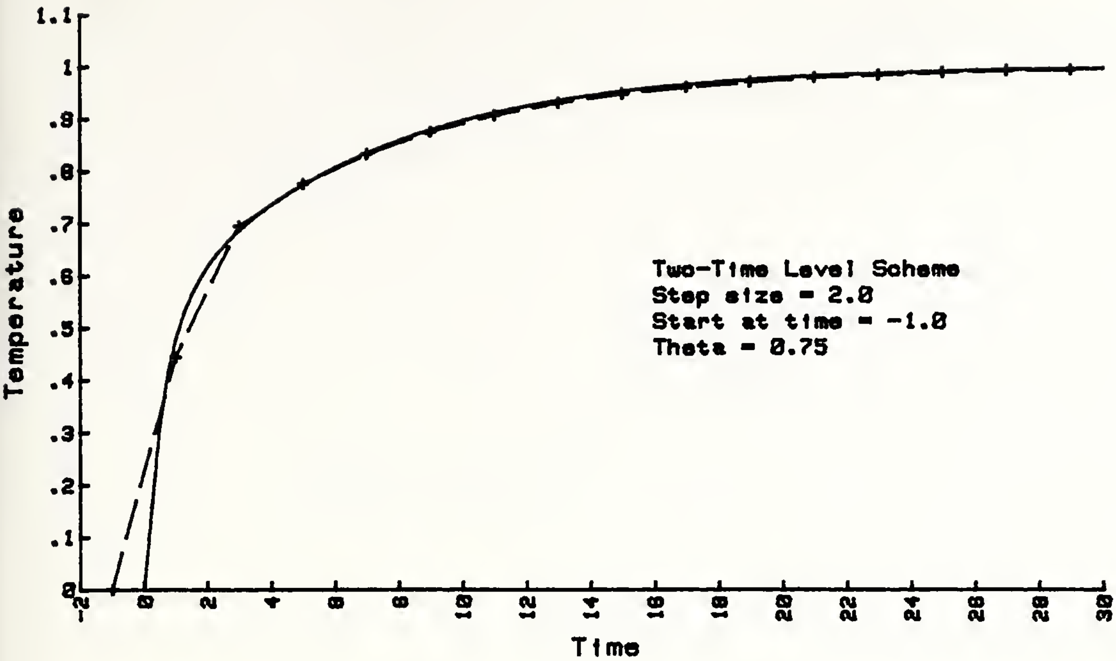
(a)



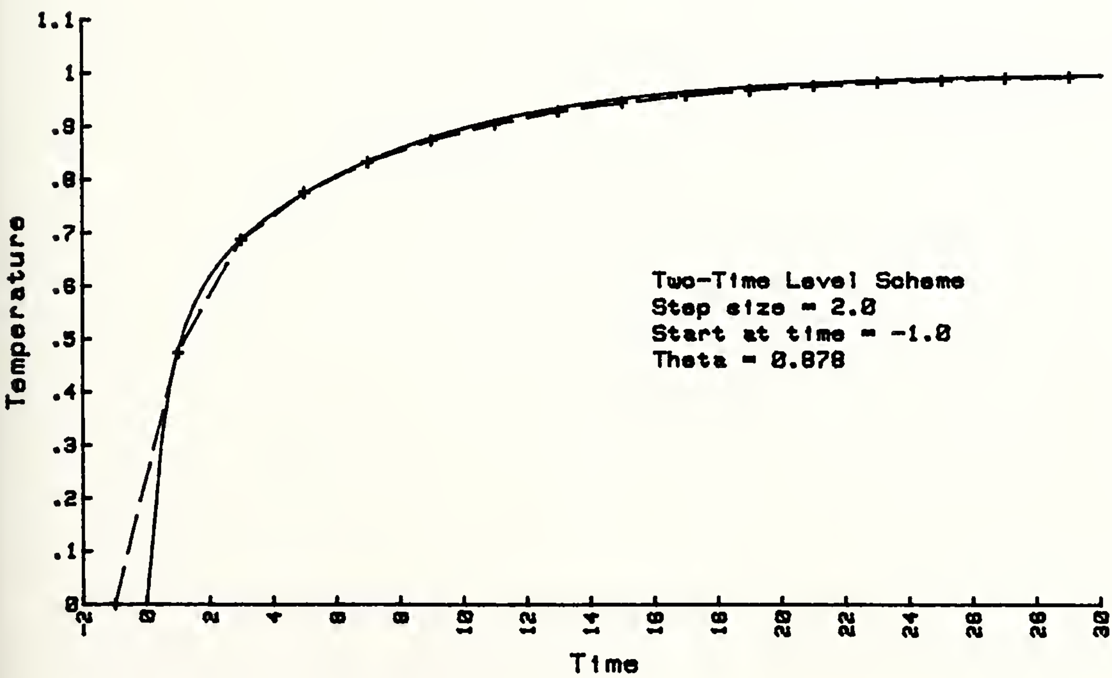
Two-Time Level Scheme
Step size = 2.0
Start at time = -1.0
Theta = 0.667
Temperature
Time

(b)

Figure 14.

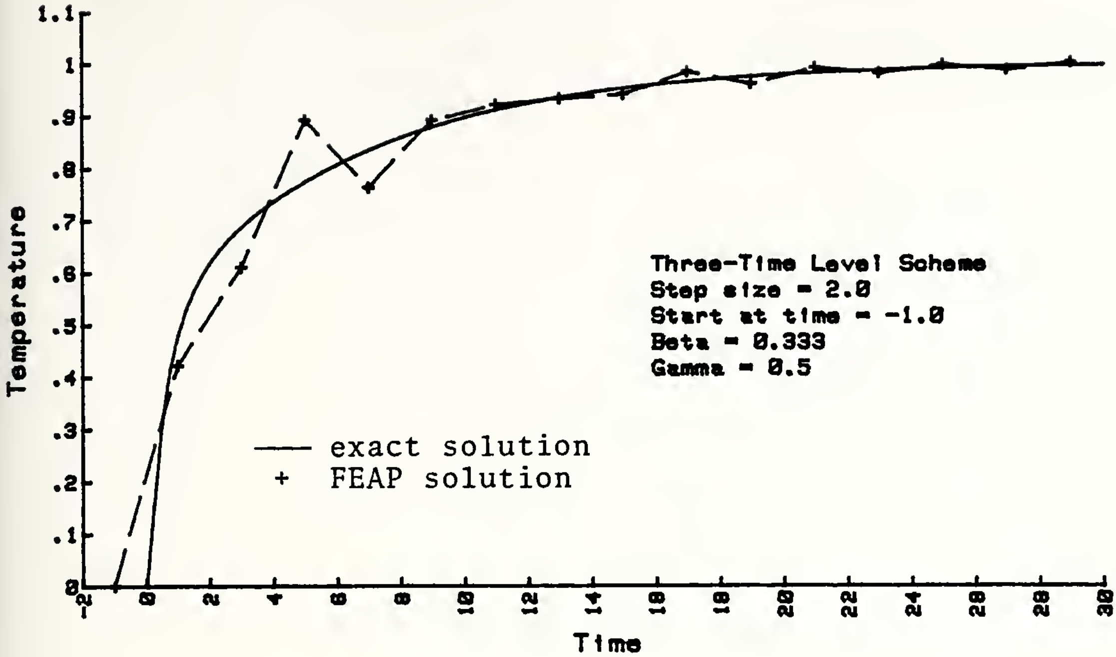


(c)



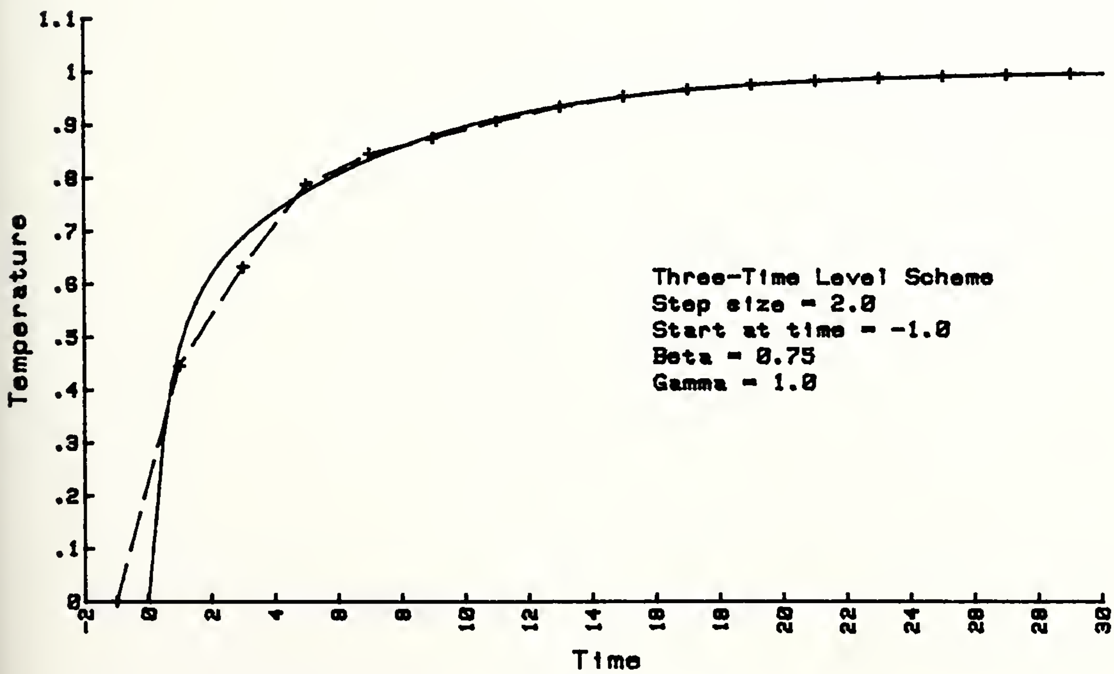
(d)

Figure 14. (continued)



Three-Time Level Scheme
Step size = 2.0
Start at time = -1.0
Beta = 0.333
Gamma = 0.5
exact solution
+ FEAP solution
Temperature
Time

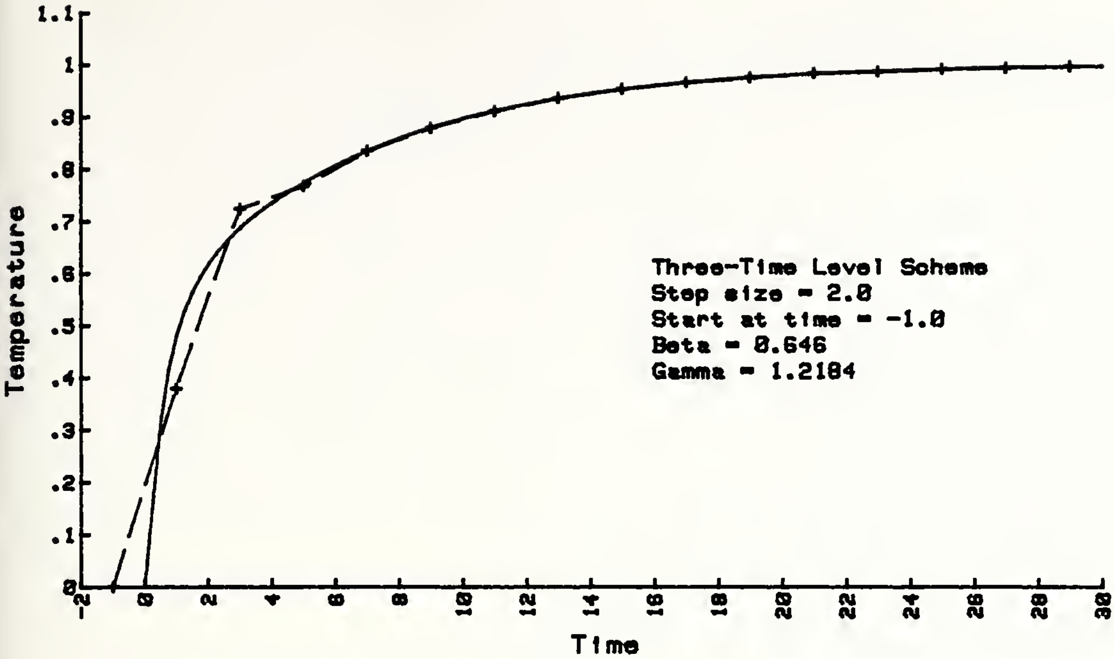
(a)



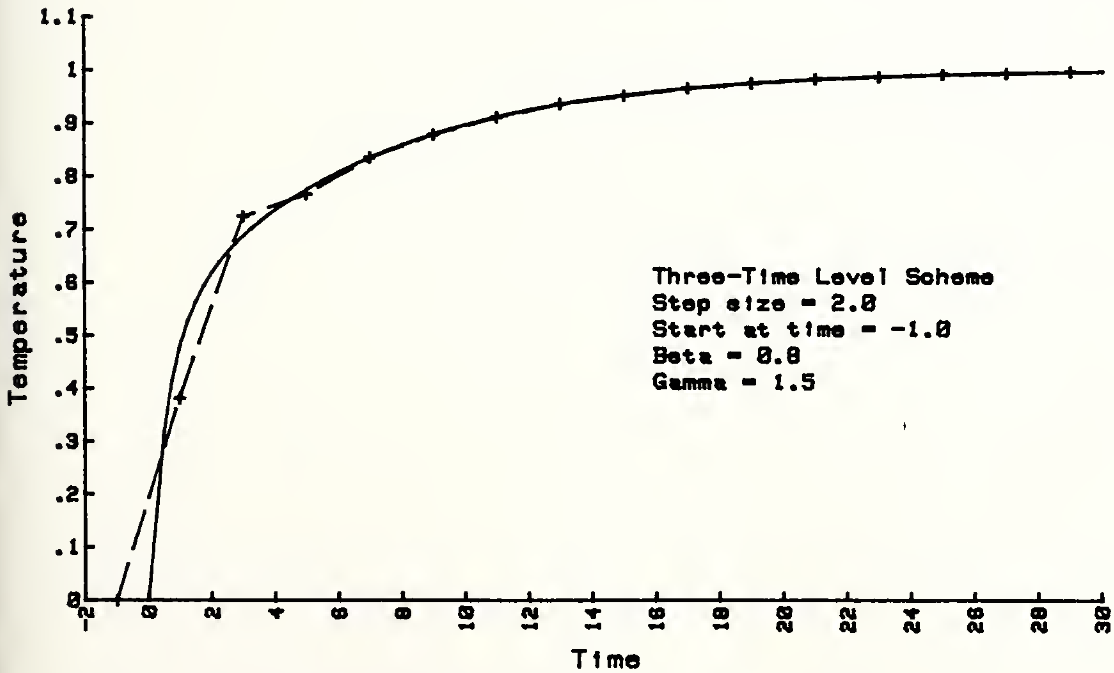
Three-Time Level Scheme
Step size = 2.0
Start at time = -1.0
Beta = 0.75
Gamma = 1.0
Temperature
Time

(b)

Figure 15.

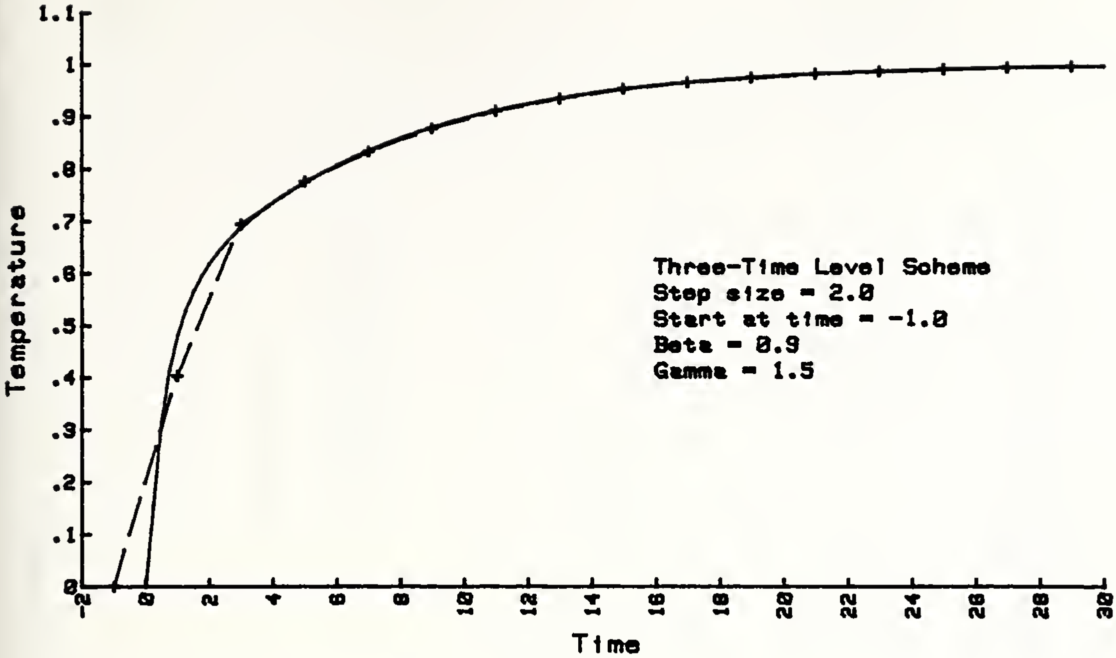


(c)

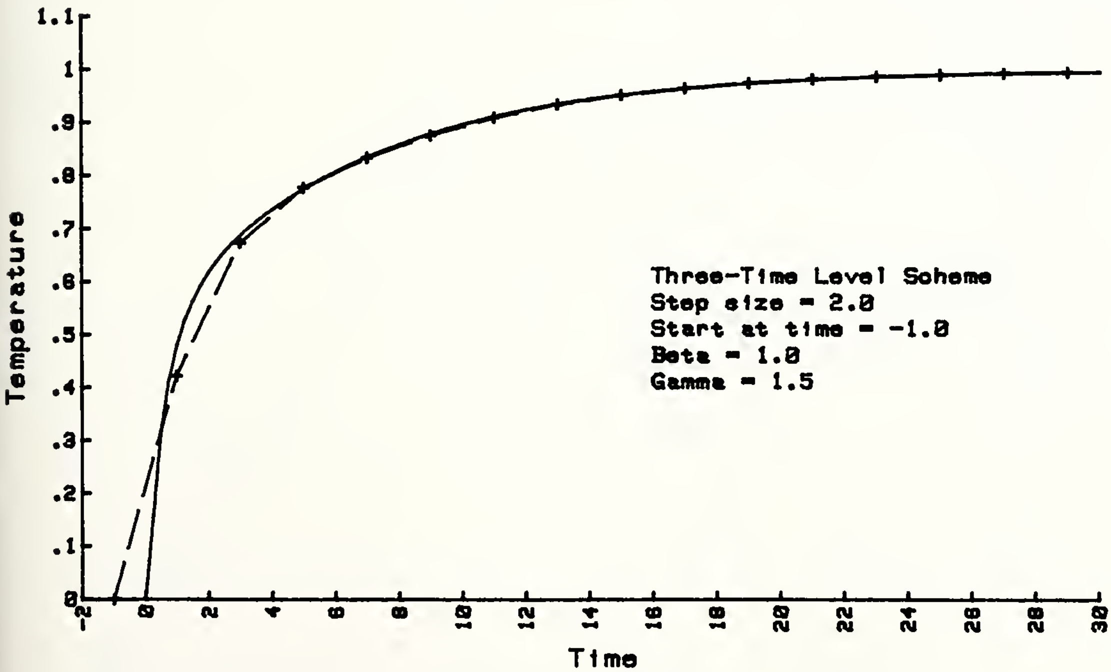


(d)

Figure 15. (continued)



(e)



(f)

Figure 15. (continued)

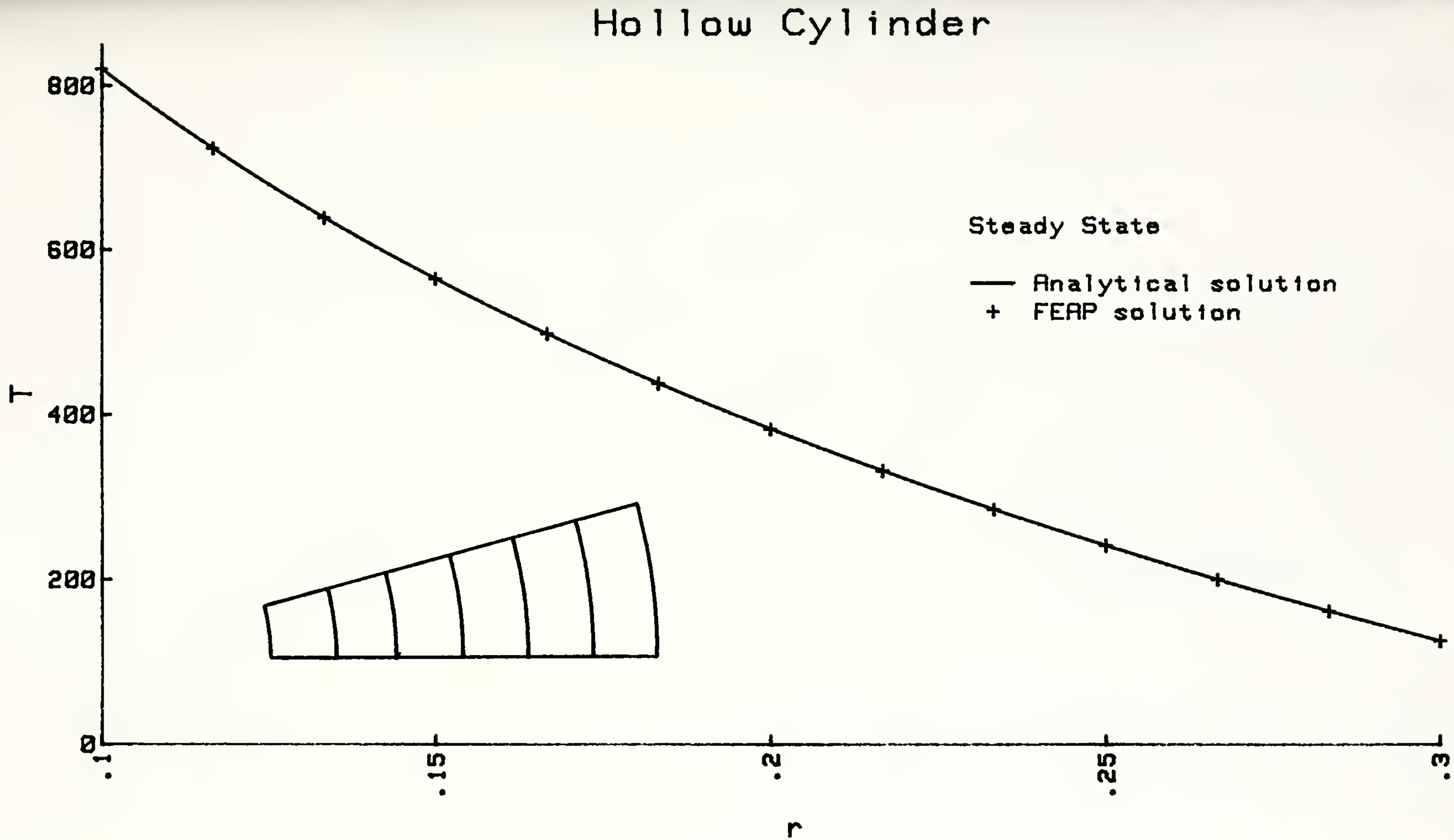


Figure 16. Radial Temperature Distribution in a Hollow Cylinder. The temperature T and the radius r are expressed in degrees Celsius and meters, respectevely.

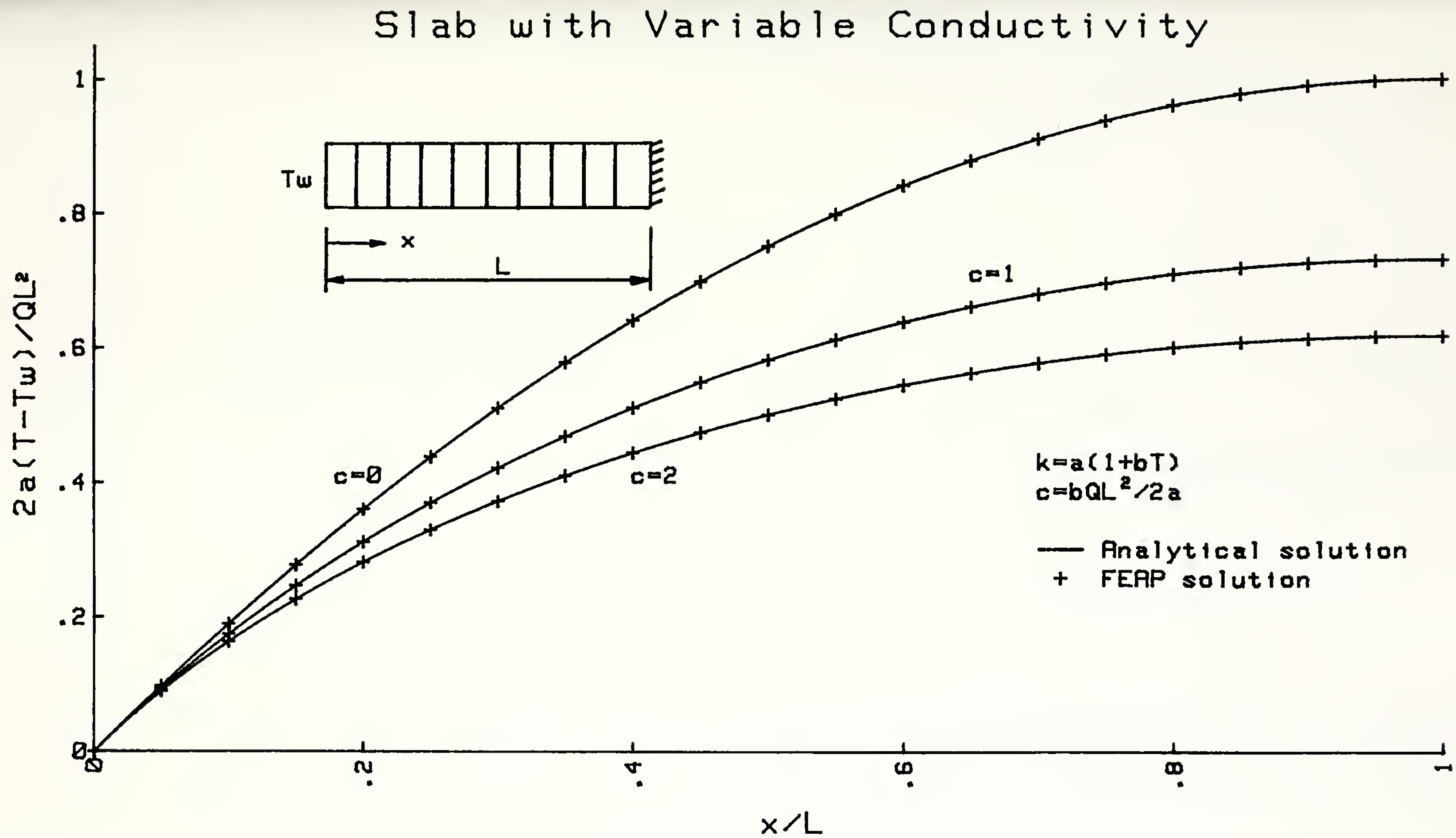


Figure 17. Effect of Linear Conductivity Variation on the Temperature of a Plate with Uniform Heat Generation.

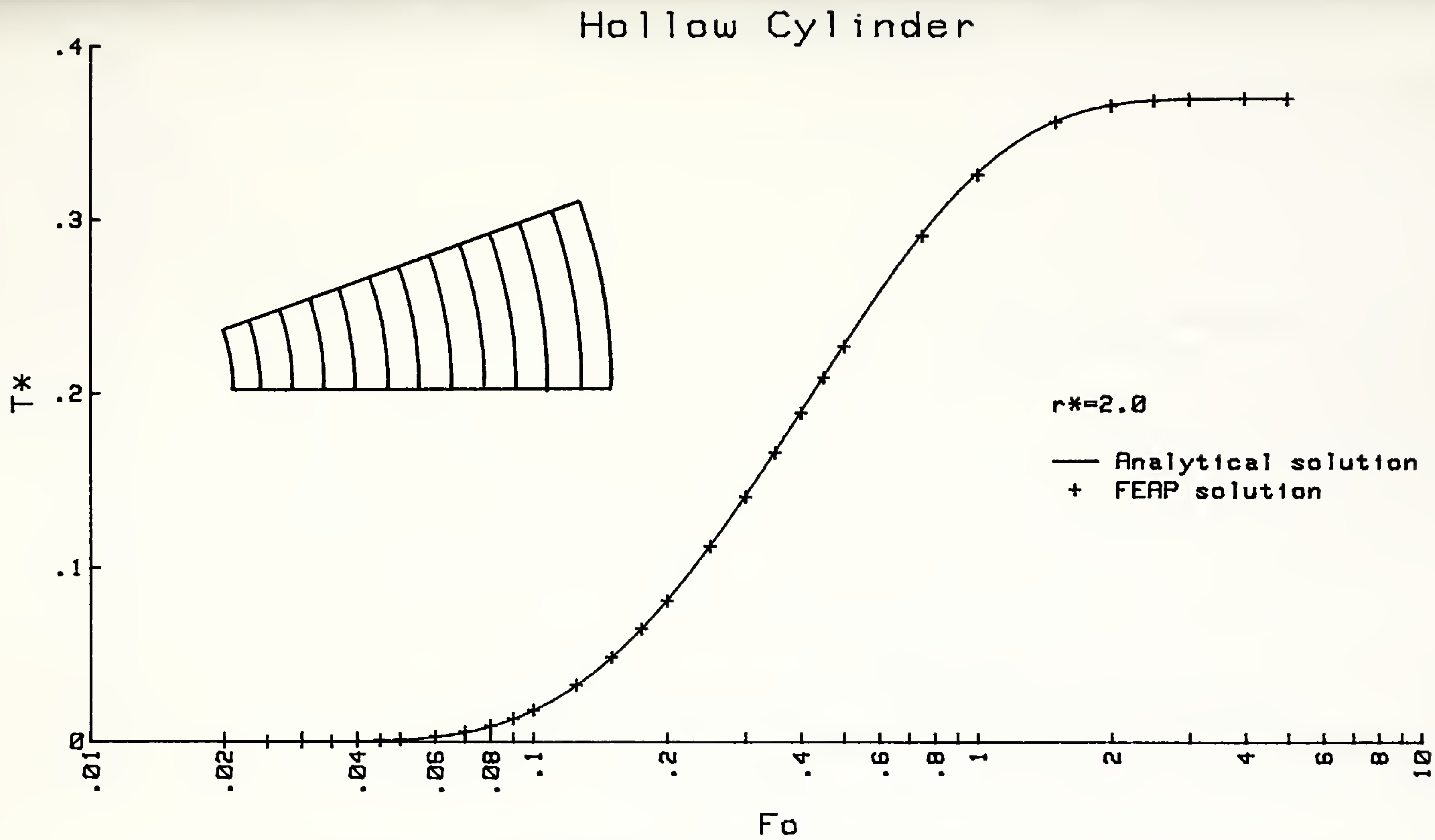


Figure 18. Unsteady Temperature of the Center Interior Points in a Hollow Cylinder.

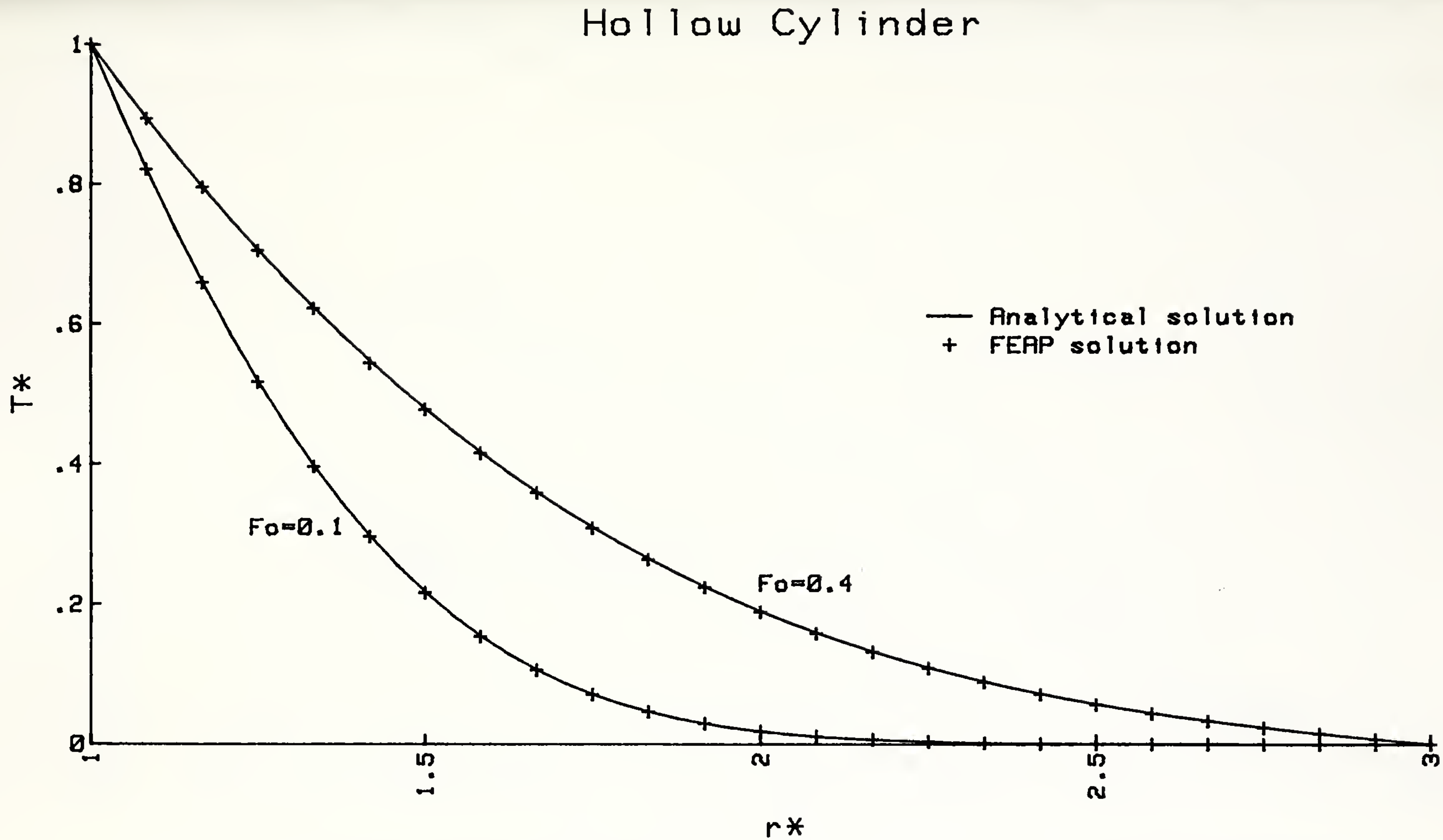


Figure 19. Unsteady Radial Temperature Distribution in a Hollow Cylinder

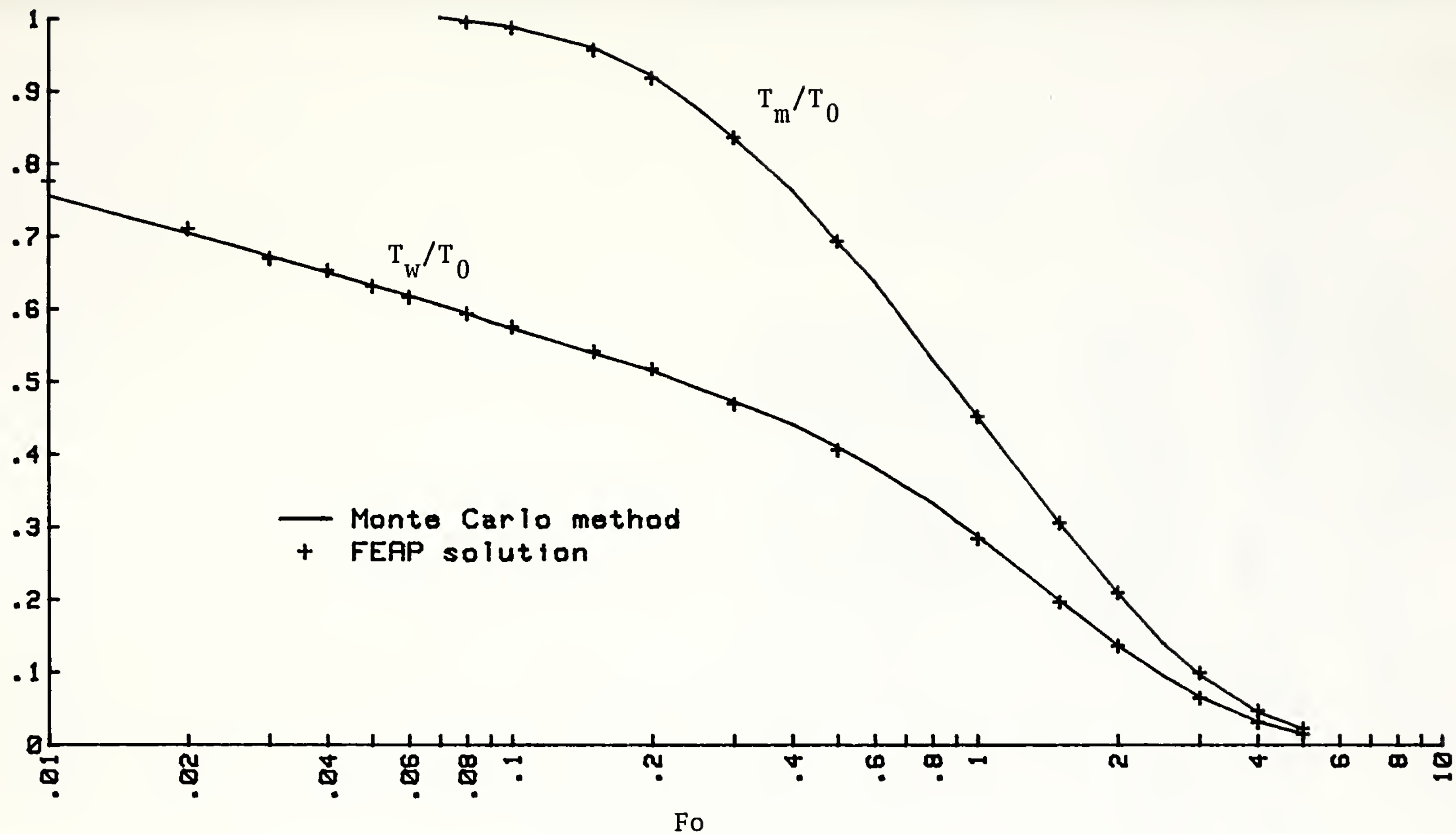


Figure 20. Unsteady Temperature Distribution in a Slab with Radiation/Convection Boundary Conditions.

```
C                       APPENDIX A
C
C              USER INSTRUCTIONS FOR FEAP
C
C      A.1 TITLE AND CONTROL INFORMATION
C
C
C.... TITLE CARD-FORMAT(20A4)
C      THE TITLE CARD ALSO SERVES AS A START OF PROBLEM CARD.
C      THE FIRST FOUR (4) COLUMNS MUST CONTAIN THE START
C      WORD FEAP
C
C        COLUMNS           DESCRIPTION          VARIABLE
C         1 TO 4      MUST CONTAIN FEAP         TITL(1)
C         5 TO 80     ALPHANUMERIC INFORMA-     TITL(I),I=2,20
C                     TION TO BE PRINTED
C                     WITH OUTPUT AS PAGE
C                     HEADER.
C
C
C.... CONTROL CARD-FORMAT(7I5)
C
C        COLUMNS           DESCRIPTION          VARIABLE
C         1 TO 5      NUMBER OF NODES           NUMNP
C         6 TO 10     NUMBER OF ELEMENTS        NUMEL
C        11 TO 15     NUMBER OF MATERIAL SETS   NUMMAT
C        16 TO 20     SPATIAL DIMENSION         NDM
C                     (MAXIMUM, UP TO 3)
C        21 TO 25     NUMBER OF UNKNOWNS        NDF
C                     PER NODE (MAXIMUM,
C                     UP TO 6)
C        26 TO 30     NUMBER OF NODES PER       NEN
C                     ELEMENT (MAXIMUM)
C        31 TO 35     NOT USED
C
```

```
C       A.2 CATA INFUT: MACRO CCNTROL STATEMENTS
C
C
C....   INPUT MACRC CCNTROL CARCS-FORMAT(A4)
C       THE INPLT CF EACH DATA SEGMENT IS CONTROLLED BY THE
C       VALUE ASSIGNEC TO CC. THE FOLLOWING VALUES ARE ADMI-
C       SSIBLE ANC EACH CC CARC MUST BE IMMEDIATELY FOLLOWED
C       BY THE APPRCPRIATE DATA.
C
C          CC VALLE          DATA TC BE INPUT
C          COOR              COORDINATE DATA
C          PCLA              CONVERT PCLAR TO CARTESIAN
C                            COCRDINATES
C          SPHE              CONVERT SPHERICAL TO CARTESIAN
C                            COORCINATES
C          ELEM              ELEMENT DATA
C          MATE              MATERIAL CATA
C          BOUN              NODAL BCUNCARY CONDITICN CATA
C          FORC              PRESCRIBED NODAL DISPLACEMENT/FORCE
C                            DATA. FCR HEAT TRANSFER PROBLEMS
C                            THIS IS PRESCRIBED NODAL TEMPERA-
C                            TURE/FLUX CATA
C          TEMP              TEMPERATURE DATA
C          PRIN              PRINT SLBSEQUENT MESH DATA
C                              (DEFAULT MODE)
C          NOPR              DO NCT PRINT SUBSEQUENT MESH CATA
C          PAGE              PRINTED OUTPUT CONTROL
C          ENC               MUST BE LAST CARD IN MESH DATA,
C                            TERMINATES MESH INPUT
C
C          EXCEPT FCR THE END CARC THE DATA SEGMENTS CAN BE
C          IN ANY CRCER. IF THE VALLES OF BOUN,FCRC, CR TEMP
C          ARE ZERC, NC INPUT CATA IS REQUIRED.
C
C
C
C
C....   COCRDINATE CATA - FORMAT(2I5,7F10.0)
C       MUST IMMECIATELY FOLLCW A COOR MACRO CARD.
C       THE CCCRCINATE CATA CARC CCNTAINS THE NODE NUMBER N
C       AND THE VALLE OF THE CCCRDINATES FOR THE NCCE. CNLY
C       THE VALUES CF (XL(I),I=1,NCM ) ARE USED, WHERE NDM IS
C       THE VALLE INPLT ON THE CCNTROL CARD.
C       NOCAL COORCINATES CAN BE GENERATED ALONG A STRAIGHT
C       LINE DESCRIBEC BY THE VALUES ON TWO SUCCESSIVE
C       CARDS. THE VALUE OF THE NCCE NUMBER IS CCMPUTEC
C       USING THE N ANC NG ON THE FIRST CARD TO COMPLLTE
C       THE SECLENCE N, N+NG,N+2*NG, ETC. NG MAY BE INPUT
C       AS A NEGATIVE NUMBER, IF IT HAS INCORRECT SIGN THE
C       SIGN WILL BE CHANGED. NOCES NEED NOT BE IN ORCER.
C
C          COLUMNS             DESCRIPTION            VARIABLE
C           1 TC  5       NCDE NUMBER                 N
C           6 TC 10       GENERATCR INCREMENT         NG
C          11 TC 20       X1 COORCINATE               XL(1)=X(1,N)
C          21 TC 30       X2 COORCINATE               XL(2)=X(2,N)
C          31 TC 40       X3 COORDINATE               XL(3)=X(3,N)
C              TERMINATE WITH BLANK CARD(S)
C
C
C
C....   PCLAR TO CARTESIAN CCNVERSION CATA-FORMAT(3I5,2F10.0)
C       MUST IMMECIATELY FOLLCW A POLA MACRO CARD.
C       THE PCLAR CCCRCINATES (R,THETA) MUST BE PREVICUSLY IN-
C       PUT AS COCRCINATE DATA (MACRC CARD COOR) WHERE R IS X1
C       ANC THETA IS X2. THE NOCAL POLAR COORDINATES CF NODES
C       NI, NI+INC, NI+2*INC,..., NE ARE CONVERTEC TC
C       CARTESIAN CCCRDINATES.
C
C          COLUMNS             DESCRIPTION            VARIABLE
C           1 TC  5       FIRST NCCE TC BE            NI
C                         CCNVERTEC
C           6 TC 10       LAST NODE TC BE             NE
```

```
C                CCNVERTEC
C       11 TC 15    NODAL INCREMENT              INC
C       16 TC 25    X COORCINATE OF THE          XO
C                   CRIGIN
C       26 TC 35    Y COORDINATE OF THE          YO
C                   CRIGIN
C           TERMINATE WITH BLANK CARD(S)
C
C
C
C.... SPHERICAL TC CART. CONVERSION DATA-FORMAT(3I5,3F10.0)
C     MUST IMMEDIATELY FOLLOW A SPHE MACRO CARD.
C     THE SPHERICAL COORDINATES (R,THETA,PHI) MUST BE PRE-
C     VIOUSLY INPUT AS COORDINATE DATA (X1,X2,X3) FOLLOWING
C     MACRO CARD COOR. THETA IS THE ANGLE FORMED BY R AND X
C      AND PHI IS THE ANGLE FORMED BY R AND Z.
C     THE NODAL SPHERICAL COORDINATES OF NODES NI, NI+INC,
C     NI+2*INC,...,NE ARE CONVERTED TO CARTESIAN COORDINATES
C
C       COLUMNS          DESCRIPTION             VARIABLE
C         1 TO 5    FIRST NODE TO BE             NI
C         6 TC 10   LAST NODE TO BE              NE
C                   CONVERTED
C       11 TC 15    NODAL INCREMENT              INC
C       16 TC 2C    X COORDINATE OF THE          XO
C                   ORIGIN
C       26 TC 35    Y COORDINATE OF THE          YO
C                   CRIGIN
C       36 TC 45    Z COORDINATE OF THE          ZO
C                   CRIGIN
C           TERMINATE WITH BLANK CARD(S).
C
C
C
C.... ELEMENT DATA - FORMAT(16I5)
C     MUST IMMEDIATELY FOLLOW AN ELEM MACRO CARD.
C     THE ELEMENT DATA CARD CONTAINS THE ELEMENT NUMBER,
C     MATERIAL SET NUMBER (WHICH ALSO SELECTS THE ELEMENT
C     TYPE,SEE MATERIAL PROPERTY DATA), AND THE SEQUENCE
C     OF NODES CONNECTED TO THE ELEMENT.IF THERE ARE LESS
C     THAN NEN NODES EITHER LEAVE THE APPROPRIATE FIELDS
C     BLANK OR PUNCH ZEROS.
C     ELEMENTS MUST BE IN ORDER. IF ELEMENT CARDS ARE
C     OMITTED THE ELEMENT DATA WILL BE GENERATED FROM
C     THE PREVIOUS ELEMENT WITH THE SAME MATERIAL NUMBER
C     AND THE NODES ALL INCREMENTED BY LX ON THE PREVIOUS
C     ELEMENT.GENERATION TO THE MAXIMUM ELEMENT NUMBER
C     OCCURS WHEN A BLANK CARD IS ENCOUNTERED.
C
C
C       COLUMNS          DESCRIPTION             VARIABLE
C         1 TC 5    ELEMENT NUMBER               L
C         6 TC 10   MATERIAL SET NUMBER          IX(NEN1,L)
C       11 TC 15    NODE 1 NUMBER                IX(1,L)
C       16 TC 20    NODE 2 NUMBER                IX(2,L)
C          ETC.          .                          .
C          ETC.     NODE NEN NUMBER              IX(NEN,L)
C          ETC.     GENERATION INCREMENT         LX
C            TERMINATE WITH BLANK CARD(S).
C
C
C
C.... MATERIAL PROPERTY DATA - FORMAT(I5,4X,I1,17A4)
C     MUST IMMEDIATELY FOLLOW A MATE MACRO CARD.
C
C       COLUMNS          DESCRIPTION             VARIABLE
C         1 TC 5    MATERIAL SET NUMBER          MA
C         6 TC 9
C              10   ELEMENT TYPE NUMBER          IEL
C       11 TC 78    ALPHANUMERIC INFORMATION     XHED
C                   TO BE OUTPUT
C       EACH MATERIAL CARD MUST BE FOLLOWED IMMEDIATELY
```

```
C         BY THE MATERIAL PROPERTY DATA REQUIRED FOR THE
C         ELEMENT TYPE IEL BEING USED,E.G.,SEE SECTION A.4
C         AND A.5 FOR HEAT TRANSFER ELEMENTS.
C
C
C
C.... BOUNDARY RESTRAINT DATA - FORMAT(16I5)
C     MUST IMMEDIATELY FOLLOW A BOUN MACRO CARD.
C     FOR EACH NODE WHICH HAS AT LEAST ONE DEGREE OF FREEDOM
C     WITH A SPECIFIED DISPLACEMENT,A BOUNDARY CONDITION
C     CARD MUST BE INPUT. THE CONVENTION USED FOR BOUNDARY
C     RESTRAINTS IS
C         .EQ.0   NO RESTRAINT,FORCE SPECIFIED
C         .NE.0   RESTRAINED.DISPLACEMENT SPECIFIED
C     VALUES OF FORCE OR DISPLACEMENT INPUT IN FORC
C
C       COLUMNS            DESCRIPTION            VARIABLE
C        1 TO 5      NODE NUMBER                  N
C        6 TO 10     GENERATION INCREMENT         NX
C       11 TO 15     DOF 1 BOUNDARY CODE          IDL(1)=ID(1,N)
C       16 TO 20     DOF 2 BOUNDARY CODE          IDL(2)=ID(2,N)
C         ETC.                 .                     .
C         ETC.       DOF NDF BOUNDARY CODE        IDL(NDF)=
C                                                      ID(NDF,N)
C       WHEN GENERATING BOUNDARY CONDITION CODES FOR
C       SUBSEQUENT NODES IDL IS SET TO ZERO IF IT WAS
C       INPUT .GE.0, AND IS SET TO -1 IF INPUT NEGA-
C       TIVE. ALL DEGREES OF FREEDOM WITH NON-ZERO
C       CODES ARE ASSUMED FIXED.
C            TERMINATE WITH BLANK CARD(S).
C
C
C
C.... NODAL FORCED BOUNDARY VALUE DATA - FORMAT(2I5,7F10.0)
C     MUST IMMEDIATELY FOLLOW A FORC MACRO CARD.
C     FOR EACH NODE WHICH HAS A NON-ZERO NODAL FORCE OR
C     DISPLACEMENT A FORCE CARD MUST BE INPUT OR GENERATED.
C     GENERATION IS THE SAME AS FOR CO-ORDINATE DATA. THE
C     VALUE SPECIFIED IS A FORCE IF THE CORRESPODING RES-
C     TRAINT CODE IS ZERO AND A DISPLACEMENT IF THE CORRES-
C     PONDING RESTRAINT CODE IS NON-ZERO.
C
C       COLUMNS            DESCRIPTION            VARIABLE
C        1 TO 5     NODE NUMBER                   N
C        6 TO 10    GENERATION INCREMENT          NG
C       11 TO 20    DOF 1 FORCE (DISP.)           XL(1)=F(1,N)
C       21 TO 30    DOF 2 FORCE (DISP.)           XL(2)=F(2,N)
C         ETC.              .
C         ETC.      DOF NDF FORCE (DISP.)         XL(NDF)=
C                                                      F(NDF,N)
C          TERMINATE WITH A BLANK CARD.
C
C
C
C.... TEMPERATURE DATA CARD - FORMAT(2I5,F10.0)
C     MUST IMMEDIATELY FOLLOW A TEMP MACRO CARD.
C     NOT USED IN HEAT TRANSFER PROBLEMS.
C     FOR EACH NODE WHICH HAS A NON-ZERO TEMPERATURE THE
C     VALUE MUST BE INPUT. GENERATION OF VALUES CAN BE
C     PERFORMED AS DESCRIBED FOR CO-ORDINATES.
C
C       COLUMNS            DESCRIPTION            VARIABLE
C        1 TO 5      NODE NUMBER                  N
C        6 TO 10     GENERATION INCREMENT         NG
C       11 TO 20     NODAL TEMPERATURE            XL(1)=T(N)
C            TERMINATE WITH BLANK CARD(S).
C
C
C
C.... PAGE CONTROL DATA - FORMAT(A1)
C     MUST IMMEDIATELY FOLLOW A PAGE MACRO CARD
C     THE VALUE IN COLUMN 1 CONTROLS THE PRINTED OUTPUT
```

```
C       ACCORDING TC THE FOLLCWING CONVENTION:
C            1 TITLE IS WRITEN AT THE TOP OF A NEW PAGE
C            0 THE PRINTED OUTPLT IS CONTINUOUS WITHOUT
C               SKIPPING PAGES (DEFAULT MODE).
```

```
C      A.3 PROBLEM SOLUTION: MACRO PROGRAMMING COMANDS
C
C
C.... MACRO PROGRAMMING COMMANDS - FORMAT(2(A4,1X),F10.0)
C      FOLLOWING IS A LIST OF MACRO INSTRUCTIONS WHICH
C      CAN BE USED TO CONSTRUCT SOLUTION ALGORITHMS. THE
C      FIRST INSTRUCTION MUST BE A CARD WITH MACR IN
C      COLUMNS 1 TO 4. THE INDICATED ISW VALUE IS USED BY
C      EACH ELEMENT ROUTINE TO PERFORM THE APPROPRIATE
C      COMPUTATIONS.
C
C    COLUMNS COLUMNS COLUMNS          DESCRIPTION
C    1 TO 4  6 TO 9  11 TO 20
C    CHEC                      PERFORM CHECK OF MESH (ISW=2)
C    CMAS                      CONSISTENT MASS FORMULATION
C                                  (ISW=5)
C    CONV                      DISPLACEMENT CONVERGENCE TEST
C    DATA                      READ DATA **MACRO COMMANDS. A
C                              **MACRO CARD IS INSERTED AS DATA
C                              FOLLOWING THE MACRO PROGRAM
C    DISP              N       PRINT NODAL DISPLACEMENTS EVERY
C                              N STEPS IN LOOP
C  **DT                V       SET TIME INCREMENT TO VALUE V
C    EIGE                      COMPUTE DOMINANT EIGENVALUE AND
C                              VECTOR OF CURRENT MASS AND
C                              SYMMETRIC STIFFNESS
C    EXCD                      EXPLICIT CENTERED DIFFERENCE
C                              INTEGRATION OF EQUATIONS OF
C                              MOTION USING LUMPED MASS. FIRST
C                              CALL RESERVES MEMORY ONLY.
C    FORM                      FORM RIGHT HAND SIDE OF EQUA-
C                              TIONS    (ISW=6)
C    LMAS                      LUMPED MASS FORMULATION (ISW=5)
C    LOOP              N       LOOP N TIMES ALL INSTRUCTIONS
C                              BETWEEN  MATCHING  NEXT
C                              INSTRUCTION.
C    MESH                      INPUT MESH CHANGES (MUST NOT
C                              CHANGE BOUNDARY RESTRAINTS).DATA
C                              FOLLOWS MACRO PROGRAM. SEE
C                              SECTION A.2.
C    NEXT                      END OF LOOP INSTRUCTION
C    ODE1    INIT              INPUT INITIAL CONDITIONS AND
C                              CONSTANTS OF INTEGRATION FOR
C                              FIRST ORDER ORDINARY DIFFEREN-
C                              TIAL EQUATION SOLUTION (DATA
C                              FOLLOWS MACRO PROGRAM)
C    ODE1    LINE              TWO-POINT INTEGRATION SCHEME
C    ODE1    QUAD              THREE-POINT INTEGRATION SCHEME
C    PROP              1       INPUT PROPORTIONAL LOAD TABLE
C                              (DATA FOLLOWS MACRO PROGRAM)
C    REAC                      COMPUTE NODAL REACTIONS(ISW=6)
C    SOLV                      SOLVE TANGENT EQUATIONS.UPDATE
C                              NODAL DISPLACEMENTS.
C    STRE              N       PRINT ELEMENT VARIABLES (E.G.,
C                              STRESSES) EVERY N STEPS IN LOOP
C                                  (ISW=4)
C    TANG                      SYMMETRIC TANGENT STIFFNESS
C                              FORMULATION (ISW=3)
C    TIME                      ADVANCE TIME BY DT VALUE
C   *TIM                       INDICATE EXECUTION TIME FOR
C                              EVERY MACRO INSTRUCTION
C  **TOL               V       SET SOLUTION CONVERGENCE TOLE-
C                              RANCE TO VALUE V
C    UTAN                      UNSYMMETRIC TANGENT STIFFNESS
C                              FORMULATION (ISW=3)
C    END                       END OF MACRO PROGRAM COMMANDS.
C                              DATA FOR PROGRAM FOLLOWS IN
C                              ORDER OF USE
C
C
C
C
C.... FIRST ORDER O.D.E. SOLVER DATA
```

```
C      MUST FCLLOW THE MACRC CARD END AND CORRESPONDS TO
C      THE MACRC INSTRUCTICN CCE1 INIT. THE FIRST CARD
C      INPUTS THE CCNTROL INFCRMATION AND THE NODAL INITIAL
C      CISPLACEMENTS ARE INPUT ON THE FOLLOWING CARDS.
C      NC AUTCMATIC TIME STEP ACJUSTMENT IS PERFORMEC IF
C      DUMAX CR CUMIN ARE LEFT BLANK OR ZERO.
C
C.... CONTRCL CARD - FORMAT(4F10.0)
C
C
C        COLUMNS          DESCRIPTION            VARIABLE
C         1 TC 1C      INTEGRATION PARAMETER     C5
C                      THETA FCR TWC PCINT
C                      SCHEME (DEFAULT VALUE
C                      THETA=2/3)
C        11 TC 2C      INTEGRATICN PARAMETER     C1
C                      GAMMA FCR THREE POINT
C                      SCHEME
C                      (DEFAULT GAMMA=1.5)
C        21 TC 3C      INTEGRATION PARAMETER     C2
C                      BETA FCR THREE PCINT
C                      SCHEME
C                      (DEFAULT BETA=.8)
C        31 TC 40      MAXIMUM CISPLACEMENT      DUMAX
C                      ALLOWED
C        41 TC 50      MINIMUM CISPLACEMENT      DUMIN
C                      ALLOWED
C
C.... INITIAL CCNCITICN DATA - FCRMAT(2I5,7F10.0)
C      FOR EACH NCDE WHICH HAS A NON-ZERO INITIAL DISPLACE-
C      MENT A CARC MUST BE INPUT CR GENERATED. THE GENERATION
C      IS THE SAME AS FOR CCCRCINATE CATA.
C
C        COLUMNS          DESCRIPTICN            VARIABLE
C         1 TC 5      NCDE NUMBER                N
C         6 TO 10     GENERATICN INCREMENT       NG
C        11 TC 10     CCF 1 CISPLACEMENT         XL(1)=U(1+N-1)
C          ETC.           .
C          ETC.       COF NDF CISPLACEMENT       XL(NCF)=
C
C            TERMINATE WITH BLANK CARD(S).
C
C
C
C.... PRCPCRTICNAL LCAC CARC - FCRMAT(2I5,6F10.0)
C      A SIMPLIFIEC PROPORTICNAL LOADING IS PERMITTEC WITH
C         PROP = A1 + A2*TIME + A3*(SIN(A4*TIME + A5))**L
C      WHERE THE CCEFFICIENTS ARE INPUT ON A CATA CARC
C      FOLLOWING THE END MACRC CARC ACCORDING TO THE
C      FCLLCWING TABLE.
C
C        COLUMN              DESCRIPTION
C         6 TC 10        L
C        11 TC 2C        MINIMUM TIME FOR WHICH PRCP IS
C                        COMPUTEC
C        21 TC 3C        MAXIMUM TIME FCR WHICH PRCP IS
C                        COMPUTED
C        31 TC 40        A1
C        41 TC 50        A2
C        51 TC 6C        A3
C        61 TC 7C        A4
C        71 TC 8C        A5
C
C
```

```
C       A.4 TWO DIMENSIONAL HEAT TRANSFER ELEMENT DATA
C
C                               +T
C                               +
C                               +
C                   4-----------7-----------3
C                   |           +           |
C                   |           +           |
C                   |           +           |
C                   |           +           |   S
C                   8           9+++++++++++6+++
C                   |                       |
C                   |                       |
C                   |                       |
C                   |                       |
C                   1-----------5-----------2
C
C       MUST FOLLOW THE MATE MACRO CARD.
C       THE TWO DIMENSIONAL HEAT TRANSFER ELEMENT IS CALLED
C       ELMT02 AND THUS THE ELEMENT TYPE NUMBER IN COLUMN 10
C       OF EACH MATERIAL SET NUMBER CARD MUST BE 2 WHEN THIS
C       ELEMENT IS REQUESTED. THE SECOND CARD GIVES GENERAL
C       INFORMATION AND IS PREPARED AS FOLLOWS:
C
C
C....   GENERAL INFORMATION DATA - FORMAT(5F10.0,4I5)
C
C         COLUMNS          DESCRIPTION              VARIABLE
C          1 TO 10    CONDUCTIVITY IN X DIR.        D(1)
C                     (IGNORED IF TEMPERATURE
C                     DEPENDENT)
C         11 TO 20    CONDUCTIVITY IN Y DIR.        D(2)
C                     (IGNORED IF TEMPERATURE
C                     DEPENDENT)
C         21 TO 30    SPECIFIC HEAT (IGNORED        D(3)
C                     IF TEMPERATURE DEPEN-
C                     DENT)
C         31 TO 40    DENSITY (IGNORED IF           D(4)
C                     TEMPERATURE DEPENDENT)
C         41 TO 50    HEAT GENERATION PER           D(5)
C                     UNIT VOLUME (IGNORED
C                     IF TEMPERATURE DEPEN-
C                     DENT)
C         51 TO 55    NUMBER OF INTEGRATION         NGP
C                     POINTS PER DIRECTION
C                      (DEFAULT 4)
C         56 TO 60    GEOMETRY TYPE                 KAT
C                     .EQ.2 FOR AXISYMMETRY
C                     .NE.2 FOR PLANE GEOMETRY
C         61 TO 65    TOTAL NUMBER OF LINES         NLBC
C                     WITH SPECIFIED BOUN-
C                     DARY CONDITIONS IN
C                     ELEMENTS WITH THE SAME
C                     MATERIAL SET NUMBER
C         66 TO 70    CODE TO INDICATE IF ANY       INOL
C                     OF THE MATERIAL PROPER-
C                     TIES IS TEMPERATURE DE-
C                     DEPENDENT.
C                     .EQ.0 IF ALL MATERIAL
C                      PROPERTIES ARE CONSTANT
C                     .NE.0 IF ANY OF THE MATE-
C                      RIAL PROPERTIES IS TEM-
C                      PERATURE DEPENDENT
C
C
C       THE FOLLOWING DATA ARE REQUIRED IF ANY OF THE MATERIAL
C       PROPERTIES IS TEMPERATURE DEPENDENT. THEIR ORDER IS
C       CRUCIAL AND THOSE CARDS WHICH CORRESPOND TO CONSTANT
C       MATERIAL PROPERTIES MUST BE OMITTED.
C
```

```
C.... MATERIAL PROPERTY CODE -FORMAT(4I5)
C     THIS CARD IS ALLWAYS REQUIRED IF INOL.NE.0 AND MUST
C     IMMEDIATELY FOLLOW THE GENERAL INFORMATION DATA CARD.
C
C       COLUMNS             DESCRIPTION             VARIABLE
C        1 TO 5        .EQ.0 CONSTANT CONDUCTI-     IKX
C                       VITY IN X DIR.
C                      .NE.0 TEMP. DEP. KX
C        6 TO 10       .EQ.0 CONSTANT CONDUCTI-     IKY
C                       VITY IN Y DIR.
C                      .NE.0 TEMP. DEP. KY
C       11 TO 15       .EQ.0 COSTANT HEAT CA        IRCC
C                       PACITY (SPECIFIC HEAT *
C                       DENSITY)
C                      .NE.0 TEMP. DEP. HEAT
C                       CAPACITY
C       16 TO 20       .EQ.0 CONSTANT HEAT GE-      IQ
C                       NERATION PER UNIT VOL.
C                      .NE.0 TEMP. DEP. Q
C
C
C.... CONDUCTIVITY IN X DIRECTION TABLE
C     THE TEMPERATURE DEPENDENT CONDUCTIVITY IN THE X
C     DIRECTION (KX) IS CALCULATED BY LINEAR INTERPOLATION
C     BETWEEN TWO CONSECUTIVE ENTRIES IN THIS TABLE. THE
C     FIRST CARD OF THIS SET OF DATA INDICATES THE NUMBER
C     OF ENTRIES IN THE TABLE. IT MUST BE FOLLOWED BY THE
C     SAME NUMBER OF CARDS WITH A PAIR OF VALUES (TEMPE-
C     RATURE AND CORRESPONDENT PROPERTY) IN EACH ONE. THEIR
C     ORDER IS CRUCIAL. THE PAIRS MUST BE ORDERED ACCORDING
C     TO THE INCREASING VALUE OF TEMPERATURE.
C     OMIT IF IKX.EQ.0
C
C     FIRST CARD -FORMAT(I5)
C
C       COLUMNS             DESCRIPTION             VARIABLE
C        1 TO 5        NUMBER OF ENTRIES OF         N
C                      THE TABLE TO BE INPUT
C
C     TABLE DATA  - FORMAT(2F10.0)
C     FOR EACH PAIR OF VALUES A CARD MUST BE INPUT. THE
C     TOTAL IS THE NUMBER SPECIFIED IN THE PRECEDING CARD.
C     LOWEST TEMPERATURE IN THE FIRST CARD, SECOND LOWEST
C     IN THE SECOND CARD, ETC.
C
C       COLUMNS             DESCRIPTION             VARIABLE
C        1 TO 10       TEMPERATURE                  XX(I)
C       11 TO 20       CONDUCTIVITY                 YY(I)
C
C
C.... CONDUCTIVITY IN Y DIRECTION TABLE
C     PREPARED IN THE SAME WAY AS THE CODUCTIVITY IN THE
C     X DIRECTION TABLE.
C     OMIT IF IKY.EQ.0
C
C
C.... HEAT CAPACITY TABLE
C     PREPARED IN THE SAME WAY AS THE CONDUCTIVITY IN THE
C     X DIRECTION TABLE.
C     OMIT IF IRCC.EQ.0
C
C
C.... HEAT GENERATION PER UNIT VOLUME TABLE
C     PREPARED IN THE SAME WAY AS THE CONDUCTIVITY IN THE
C     X DIRECTION TABLE.
C     OMIT IF IQ.EQ.0
C
C
C.... LINE BOUNDARY CONDITION DATA - FORMAT(3I5,2F10.0)
C     OMIT IF NLBC.EQ.0.
C     A CARD MUST INPUT FOR EACH LINE BOUNDARY CONDITION.
C     IF THE SAME LINE IS SUBJECTED TO MORE THAN ONE TYPE
```

```
C       OF BOUNDARY CONDITION, A CARD MUST BE USED FOR EACH
C       ONE OF THESE TYPES.
C       THE TOTAL NUMBER OF CARDS MUST BE NLBC.
C       THE PROPERTY VALUE IS DEFINED:
C       FOR CONVECTION - CONSTANT HEAT TRANSFER COEFFICIENT
C         (IGNORED IF KBCOND(I).EQ.4)
C       FOR FLUX - FLUX PER UNIT AREA
C       FOR RADIATION - PRODUCT OF EMISSIVITY BY STEFAN-
C                           BOLTZMAN CONSTANT
C       THE AMBIENT TEMPERATURE IS IGNORED FOR FLUX BOUNDARY
C       CONDITION
C
C          COLUMNS            DESCRIPTION             VARIABLE
C            1 TO  5       ELEMENT NUMBER             KEL(I)
C            6 TO 10       CODE TO INDICATE BOUN-     KBCOND(I)
C                          DARY CONDITION TYPE
C                          .EQ.1 FLUX
C                          .EQ.2 CONVECTION (CONS-
C                           TANT COEFFICIENT)
C                          .EQ.3 RADIATION
C                          .EQ.4 CONVECTION (TEMP.
C                           DEP. COEFF.)
C           11 TO 15       CODE TO INDICATE LINE      KLINE(I)
C                          .EQ. 1   S=+1 LINE
C                          .EQ.-1   S=-1 LINE
C                          .EQ. 2   T=+1 LINE
C                          .EQ.-2   T=-1 LINE
C           16 TO 25       PROPERTY VALUE             PROPB(I,1)
C           26 TO 35       AMBIENT TEMPERATURE        PROPB(I,2)
C
C
```

```
C      A.5 THREE DIMENSIONAL HEAT TRANSFER ELEMENT DATA
C
C                    3-----------10------------2
C                   /.                        /|
C                  / .                       / |
C                 /  .          +T          /  |
C             11 /   .          +        9 /   |
C               /   19          +         /   18
C              /     .          +        /     |
C             /      .          +       /      |
C            /       .          +      /       |
C           4--------.--12--------------1   S  |
C           |        .     21++++++|++++++     |
C           |        7......+...14..|.........6
C           |       .     +         |        /
C           |      .     +          |       /
C          20     .     +          17      /
C           |   15     +            |     13
C           |   .     +             |     /
C           |  .    R+              |    /
C           | .                     |   /
C           |.                      |  /
C           8-----------16----------5
C
C      MUST FOLLOW THE MATE MACRO CARD.
C      THE THREE DIMENSIONAL HEAT TRANSFER ELEMENT IS CALLED
C      ELMT03 AND THUS THE ELEMENT TYPE NUMBER IN COLUMN 10
C      OF EACH MATERIAL SET NUMBER CARD MUST BE 3 WHEN THIS
C      ELEMENT IS REQUESTED. THE SECOND CARD GIVES GENERAL
C      INFORMATION AND IS PREPARED AS FOLLOWS:
C
C
C.... GENERAL INFORMATION DATA - FORMAT(6F10.0,4I5)
C
C        COLUMNS          DESCRIPTION              VARIABLE
C         1 TO 10      CONDUCTIVITY IN X DIR.      D(1)
C                      (IGNORED IF TEMPERATURE
C                      DEPENDENT)
C        11 TO 20      CONDUCTIVITY IN Y DIR.      D(2)
C                      (IGNORED IF TEMPERATURE
C                      DEPENDENT)
C        21 TO 30      CONDUCTIVITY IN Z DIR.      D(3)
C                      (IGNORED IF TEMPERATURE
C                      DEPENDENT)
C        31 TO 40      SPECIFIC HEAT (IGNORED      D(4)
C                      IF TEMPERATURE DEPEN-
C                      DENT)
C        41 TO 50      DENSITY (IGNORED IF         D(5)
C                      TEMPERATURE DEPENDENT)
C        51 TO 60      HEAT GENERATION PER         D(6)
C                      UNIT VOLUME (IGNORED
C                      IF TEMPERATURE DEPEN-
C                      DENT)
C        61 TO 65      NUMBER OF INTEGRATION       NGP
C                      POINTS PER DIRECTION
C                       (DEFAULT 4)
C        66 TO 70      GEOMETRY TYPE               KAT
C                      .EQ.2 FOR AXISYMMETRY
C                      .NE.2 FOR PLANE GEOMETRY
C        71 TO 75      TOTAL NUMBER OF SURFACES    NSBC
C                      WITH SPECIFIED BOUN-
C                      DARY CONDITIONS IN
C                      ELEMENTS WITH THE SAME
C                      MATERIAL SET NUMBER
C        76 TO 80      CODE TO INDICATE IF ANY     INDL
C                      OF THE MATERIAL PROPER-
C                      TIES IS TEMPERATURE DE-
C                      DEPENDENT.
C                      .EQ.0 IF ALL MATERIAL
C                       PROPERTIES ARE CONSTANT
```

```
C                         .NE.0 IF ANY OF THE MATE-
C                          RIAL PRCPERTIES IS TEM-
C                          PERATURE DEPENDENT
C
C
C     THE FOLLOWING DATA ARE REQUIRED IF ANY OF THE MATERIAL
C     PRCPERTIES IS TEMPERATURE DEPENDENT. THEIR ORDER IS
C     CRUCIAL AND THOSE CARDS WHICH CORRESPOND TC CONSTANT
C     MATERIAL PRCPERTIES MUST BE OMITTED.
C
C.... MATERIAL PROPERTY CODE -FORMAT(4I5)
C     THIS CARD IS ALLWAYS REQUIRED IF INOL.NE.0 AND MUST
C     IMMEDIATELY FCLLOW THE GENERAL INFORMATION DATA CARD.
C
C       COLUMNS           DESCRIPTION             VARIABLE
C        1 TC 5      .EQ.0 CONSTANT CONDUCTI-     IKX
C                     VITY IN X DIR.
C                    .NE.0 TEMP. DEP. KX
C        6 TC 10     .EQ.0 CONSTANT CONDUCTI-     IKY
C                     VITY IN Y DIR.
C                    .NE.0 TEMP. DEP. KY
C       11 TC 15     .EQ.0 CONSTANT CONDUCTI-     IKZ
C                     VITY IN Z DIR.
C                    .NE.0 TEMP. DEP. KZ
C       16 TC 20     .EQ.0 COSTANT HEAT CA        IROC
C                     PACITY (SPECIFIC HEAT *
C                     DENSITY)
C                    .NE.0 TEMP. DEP. HEAT
C                     CAPACITY
C       21 TC 25     .EQ.0 CONSTANT HEAT GE-      IQ
C                     NERATICN PER UNIT VOL.
C                    .NE.0 TEMP. DEP. Q
C
C
C.... CONDUCTIVITY IN X DIRECTION TABLE
C     THE TEMPERATURE DEPENDENT CONDUCTIVITY IN THE X
C     DIRECTION (KX) IS CALCULATED BY LINEAR INTERPOLATION
C     BETWEEN TWC CONSECUTIVE ENTRIES IN THIS TABLE. THE
C     FIRST CARD CF THIS SET CF DATA INDICATES THE NUMBER
C     CF ENTRIES IN THE TABLE. IT MUST BE FOLLOWED BY THE
C     SAME NUMBER CF CARDS WITH A PAIR OF VALUES (TEMPE-
C     RATURE AND CORRESPONDENT PROPERTY) IN EACH ONE. THEIR
C     CRDER IS CRUCIAL. THE PAIRS MUST BE ORDERED ACCORDING
C     TC THE INCREASING VALUE CF TEMPERATURE.
C     OMIT IF IKX.EQ.0
C
C     FIRST CARD -FORMAT(I5)
C
C       COLUMNS           DESCRIPTION             VARIABLE
C        1 TC 5      NUMBER CF ENTRIES CF         N
C                    THE TABLE TC BE INPUT
C
C     TABLE DATA   - FORMAT(2F10.0)
C     FOR EACH PAIR OF VALUES A CARD MUST BE INPUT. THE
C     TOTAL IS THE NUMBER SPECIFIED IN THE PRECEDING CARD.
C     LOWEST TEMPERATURE IN THE FIRST CARD, SECOND LOWEST
C     IN THE SECOND CARD, ETC.
C
C       COLUMNS           DESCRIPTICN             VARIABLE
C        1 TC 10     TEMPERATURE                  XX(I)
C       11 TC 20     CONDUCTIVITY                 YY(I)
C
C
C.... CONDUCTIVITY IN Y DIRECTION TABLE
C     PREPARED IN THE SAME WAY AS THE CODUCTIVITY IN THE
C     X DIRECTICN TABLE.
C     OMIT IF IKY.EQ.0
C
C
C.... CONDUCTIVITY IN Z DIRECTION TABLE
C     PREPARED IN THE SAME WAY AS THE CODUCTIVITY IN THE
C     X DIRECTICN TABLE.
```

```
C       OMIT IF IKZ.EQ.0
C
C
C.....  HEAT CAPACITY TABLE
C       PREPARED IN THE SAME WAY AS THE CONDUCTIVITY IN THE
C       X DIRECTION TABLE.
C       OMIT IF IRCC.EQ.0
C
C
C.....  HEAT GENERATION PER UNIT VOLUME TABLE
C       PREPARED IN THE SAME WAY AS THE CONDUCTIVITY IN THE
C       X DIRECTION TABLE.
C       OMIT IF IQ.EQ.0
C
C
C.....  SURFACE BOUNDARY CONDITION DATA - FORMAT(3I5,2F10.0)
C       OMIT IF NSBC.EQ.0.
C       A CARD MUST INPUT FOR EACH SURFACE BOUNDARY CONDITION.
C       IF THE SAME SURFACE IS SUBJECTED TO MORE THAN ONE TYPE
C       OF BOUNDARY CONDITION, A CARD MUST BE USED FOR EACH
C       ONE OF THESE TYPES.
C       THE TOTAL NUMBER OF CARDS MUST BE NSBC.
C       THE PROPERTY VALUE IS DEFINED:
C       FOR CONVECTION - CONSTANT HEAT TRANSFER COEFFICIENT
C         (IGNORED IF KBCOND(I).EQ.4)
C       FOR FLUX - FLUX PER UNIT AREA
C       FOR RADIATION - PRODUCT OF EMISSIVITY BY STEFAN-
C                       BOLTZMAN CONSTANT
C       THE AMBIENT TEMPERATURE IS IGNORED FOR FLUX BOUNDARY
C       CONDITION
C
C          COLUMNS          DESCRIPTION               VARIABLE
C           1 TO  5     ELEMENT NUMBER                KEL(I)
C           6 TO 10     CODE TO INDICATE BOUN-        KBCOND(I)
C                       DARY CONDITION TYPE
C                       .EQ.1 FLUX
C                       .EQ.2 CONVECTION (CONS-
C                        TANT COEFFICIENT)
C                       .EQ.3 RADIATION
C                       .EQ.4 CONVECTION (TEMP.
C                        DEP. COEFF.)
C          11 TO 15     CODE TO INDICATE SURF.        KSURF(I)
C                       .EQ. 1   R=+1 SURF
C                       .EQ.-1   R=-1 SURF
C                       .EQ. 2   S=+1 SURF
C                       .EQ.-2   S=-1 SURF
C                       .EQ. 3   T=+1 SURF
C                       .EQ.-3   T=-1 SURF
C          16 TO 25     PROPERTY VALUE                PROPB(I,1)
C          26 TO 35     AMBIENT TEMPERATURE           PROPB(I,2)
C
```

APPENDIX B

SAMPLE EXAMPLE

B.1 DATA CARDS

```
FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER
   33     6     1     2     1     8
COOR
    1     5   .1
   31         .3
    4     5.116666667
   29      .283333333
    2     5   .1          3.
   32         .3          3.
    3     5   .1          6.
   33         .3          6.
    5     5.116666667     6.
   30      .283333333     6.

 POLA
    1    33     1

 ELEM
    1     1     1     6     8     3     4     7     5     2     5

 BOUN
    1     1    -1
    3          -1

 FORC
    1     1   820.
    3         820.
```

```
MATE
    1     2
10.             10.                                        3          1
    6     2     1  200.       20.
END
MACR
TANG
FORM
SOLV
DISP
END
STOP
```

B.2 COMPUTER OUTPUT

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

```
NUMBER OF NODAL POINTS        =    33
NUMBER OF ELEMENTS            =     6
NUMBER OF MATERIAL SETS       =     1
DIMENSION OF COORDINATE SPACE=      2
DEGREES OF FREEDOM/NODE       =     1
NODES PER ELEMENT (MAXIMUM)   =     8
EXTRA D.O.F. TO ELEMENT       =     0
```

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

NODAL COORDINATES

NODE	1 COORD	2 COORD
1	0.1000	0.0
2	0.1000	3.0000
3	0.1000	6.0000
4	0.1167	0.0
5	0.1167	6.0000
6	0.1333	0.0
7	0.1333	3.0000
8	0.1333	6.0000
9	0.1500	0.0
10	0.1500	6.0000
11	0.1667	0.0
12	0.1667	3.0000
13	0.1667	6.0000
14	0.1833	0.0
15	0.1833	6.0000
16	0.2000	0.0
17	0.2000	3.0000

18	0.2000	6.0000
19	0.2167	0.0
20	0.2167	6.0000
21	0.2333	0.0
22	0.2333	3.0000
23	0.2333	6.0000
24	0.2500	0.0
25	0.2500	6.0000
26	0.2667	0.0
27	0.2667	3.0000
28	0.2667	6.0000
29	0.2833	0.0
30	0.2833	6.0000
31	0.3000	0.0
32	0.3000	3.0000
33	0.3000	6.0000

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

CARTESIAN COORDINATES COMPUTED FROM POLAR INPUT WITH XO = 0.0 YO = 0.0

NODE	1-COORD	2-COORD
1	0.1000	0.0
2	0.0999	0.0052
3	0.0995	0.0105
4	0.1167	0.0
5	0.1160	0.0122
6	0.1333	0.0
7	0.1332	0.0070
8	0.1326	0.0139
9	0.1500	0.0
10	0.1492	0.0157
11	0.1667	0.0
12	0.1664	0.0087
13	0.1658	0.0174
14	0.1833	0.0

15	0.1823	0.0192
16	0.2000	0.0
17	0.1997	0.0105
18	0.1989	0.0209
19	0.2167	0.0
20	0.2155	0.0226
21	0.2333	0.0
22	0.2330	0.0122
23	0.2321	0.0244
24	0.2500	0.0
25	0.2486	0.0261
26	0.2667	0.0
27	0.2663	0.0140
28	0.2652	0.0279
29	0.2833	0.0
30	0.2818	0.0296
31	0.3000	0.0
32	0.2996	0.0157
33	0.2984	0.0314

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

ELEMENTS

ELEMENT	MATERIAL	1 NODE	2 NODE	3 NODE	4 NODE	5 NODE	6 NODE	7 NODE	8 NODE
1	1	1	6	8	3	4	7	5	2
2	1	6	11	13	8	9	12	10	7
3	1	11	16	18	13	14	17	15	12
4	1	16	21	23	18	19	22	20	17
5	1	21	26	28	23	24	27	25	22
6	1	26	31	33	28	29	32	30	27

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

NODAL B.C.

```
NODE      1 B.C.
   1          -1
   2          -1
   3          -1

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

    NODAL FORCE/DISPL

     NODE       1 FORCE
        1     820.0000
        2     820.0000
        3     820.0000

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

    MATERIAL PROPERTIES

    MATERIAL SET  1 FOR ELEMENT TYPE 2

         DEGREE OF FREEDOM ASSIGNMENTS    LOCAL    GLOBAL
                                          NUMBER   NUMBER
                                            1        1
    HEAT CONDUCTION ELEMENT

      CONDUCTIVITY     KX =    10.00000         KY =   10.00000
      SPECIFIC HEAT        0.0
      DENSITY              0.0
      HEAT GENER/UNIT VOL  0.0

        3  GAUSS PTS/DIR
        1  LINES WITH SPECIFIED BOUNDARY CONDITIONS
      PLANE   ANALYSIS
```

```
FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

    LINE B.C.

     ELEM     B.C.    LINE  PROPERTY VALUE     TEMPERATURE
        6       2       1      200.0000          20.00000

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

    MACRO INSTRUCTIONS

    MACRO STATEMENT      VARIABLE 1      VARIABLE 2
         TANG           0.0             0.0
         FORM           0.0             0.0
         SOLV           0.0             0.0
         DISP           0.0             0.0
         END            0.0             0.0
 **MACRO INSTRUCTION    1 START EXECUTION**  TANG          V1 =    0.0         , V2 =    0.0
 **MACRO INSTRUCTION    2 START EXECUTION**  FORM          V1 =    0.0         , V2 =    0.0
    FORCE CONVERGENCE TEST
         RNMAX =      5911.5          RN     =      5911.5          TOL    =    0.10000D-08
 **MACRO INSTRUCTION    3 START EXECUTION**  SOLV          V1 =    0.0         , V2 =    0.0
    ENERGY (DR*A*DR) =  0.4753833140D 07
 **MACRO INSTRUCTION    4 START EXECUTION**  DISP          V1 =    0.0         , V2 =    0.0

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

    PROPORTIONAL LOAD   1.0000

FEAP * * RADIAL TEMPERATURE DISTRIBUITION IN A HOLLOW CYLINDER

    NODAL DISPLACEMENTS     TIME  0.0

     NODE      1 COORD      2 COORD       1 DISPL
        1       0.1000       0.0      0.82000D 03
```

2	0.0999	0.0052	0.82000D 03
3	0.0995	0.0105	0.82000D 03
4	0.1167	0.0	0.72256D 03
5	0.1160	0.0122	0.72256D 03
6	0.1333	0.0	0.63811D 03
7	0.1332	0.0070	0.63811D 03
8	0.1326	0.0139	0.63811D 03
9	0.1500	0.0	0.56365D 03
10	0.1492	0.0157	0.56365D 03
11	0.1667	0.0	0.49702D 03
12	0.1664	0.0087	0.49702D 03
13	0.1658	0.0174	0.49702D 03
14	0.1833	0.0	0.43677D 03
15	0.1823	0.0192	0.43677D 03
16	0.2000	0.0	0.38175D 03
17	0.1997	0.0105	0.38175D 03
18	0.1989	0.0209	0.38175D 03
19	0.2167	0.0	0.33114D 03
20	0.2155	0.0226	0.33114D 03
21	0.2333	0.0	0.28428D 03
22	0.2330	0.0122	0.28428D 03
23	0.2321	0.0244	0.28428D 03
24	0.2500	0.0	0.24066D 03
25	0.2486	0.0261	0.24066D 03
26	0.2667	0.0	0.19985D 03
27	0.2663	0.0140	0.19985D 03
28	0.2652	0.0279	0.19985D 03
29	0.2833	0.0	0.16152D 03
30	0.2818	0.0296	0.16152D 03
31	0.3000	0.0	0.12538D 03
32	0.2996	0.0157	0.12538D 03
33	0.2984	0.0314	0.12538D 03

END OF MACRO EXECUTION

APPENDIX C

FEAP LISTING

```
C       MAIN PROGRAM
C
C.....  FINITE ELEMENT ANALYSIS PROGRAM (FEAP) FOR SOLUTION OF GENERAL
C.....  PROBLEM CLASSES USING THE FINITE ELEMENT METHOD.  PROBLEM SIZE
C.....  IS CONTROLLED BY THE DIMENSION IN BLANK COMMON AND THE VALUE OF
C.....  MAX AS SET IN THE MAIN PROGRAM.
C
C.....  PROGRAMMED BY PROF. R.L.TAYLOR, DEPARTMENT OF CIVIL ENGINEERING,
C.....  UNIVERSITY OF CALIFORNIA, BERKELEY, CALIFORNIA 94720, U.S.A.
C
C.....  EXPANDED BY J.M.BETTENCOURT, DEPARTMENT OF MECHANICAL ENGINEERING,
C.....  NAVAL POSTGRADUATE SCHOOL, MONTEREY, CA 93940
C
C.....  SET PROGRAM CAPACITY
        COMMON M(4000)
        COMMON /PSIZE/ MAX
        MAX = 4000
        CALL PCONTR
        STOP
        END


        SUBROUTINE PCONTR
        IMPLICIT REAL*8 (A-H,O-Z)
C
        COMMON M(1)
        COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
        COMMON /LABEL/ PDIS(6),A(6),BC(2),DI(6),CD(3),TE(3),FD(3)
        COMMON /MDATA/ N01,N0,N1,N2,N3,N4,N5,N6,N7,N8,N9,N10,N11,N12,N13
        LOGICAL PCOMP
        DIMENSION TITL(20),WD(3)
        DATA WD/4HFEAP,4HMACR,4HSTOP/
C.....  READ A CARD AND COMPARE FIRST FOUR COLUMNS WITH MACRO LIST
1       READ(5,1000) TITL
        IF(PCOMP(TITL(1),WD(1))) GO TO 100
        IF(PCOMP(TITL(1),WD(2))) GO TO 200
        IF(PCOMP(TITL(1),WD(3))) RETURN
        GO TO 1
C.....  READ AND PRINT CONTROL INFORMATION
100     DO 101 I = 1,20
101     HEAD(I) = TITL(I)
        READ(5,1001) NUMNP,NUMEL,NUMMAT,NDM,NDF,NEN,NAD
        WRITE(6,2000) HEAD,NUMNP,NUMEL,NUMMAT,NDM,NDF,NEN,NAD
C.....  SET POINTERS FOR ALLOCATION OF DATA ARRAYS
```

```
      PDIS(2) = A(NDM)
      NEN1 = NEN + 1
      NST = NEN*NDF + NAD
      N01 = 1 + NUMNP*NDF*IPR
      N01 = N01 + MOD((N01-1),IPR)
      N0 = N01 + NST*2*IPR
      N1 = N0 + NEN*NDM*IPR
      N2 = N1 + NEN*IPR
      N3 = N2 + NST
      N3 = N3 + MOD((N3-1),IPR)
      N4 = N3 + NST*IPR
      N5 = N4 + NST*NST*IPR
      N6 = N5 + NUMMAT*7
      N6 = N6 + MOD((N6-1),IPR)
      N7 = N6 + 10*NUMMAT*IPR
      N8 = N7 + NDF*NUMNP
      N8 = N8 + MOD((N8-1),IPR)
      N9 = N8 + NDM*NUMNP*IPR
      N10 = N9 + NEN1*NUMEL
      N10 = N10 + MOD((N10-1),IPR)
      N11 = N10 + NDF*NUMNP*IPR
      N12 = N11 + NUMNP*IPR
      N13 = N12 + NDF* NUMNP
C.... CHECK THAT SUFFICIENT MEMORY EXISTS
      CALL SETMEM(N13)
      CALL PZERO(M,N13)
C.... CALL MESH INPUT SUBROUTINE TO READ AND PRINT ALL MESH DATA
      III = 0
      CALL PMESH(M(N2),M(N5),M(N6),M(N7),M(N8),M(N9),M(N10),M(N11),NDF,
     1   NDM,NEN1,III)
C.... ESTABLISH PROFILE OF RESULTING EQUATIONS FOR STIFFNESS, MASS, ETC.
      CALL PROFIL(M(N12),M(N7),M(N9),NDF,NEN1,NAD)
C.... SET POINTERS FOR SOLUTION ARRAYS * CHECK FOR SUFFICIENT MEMORY
      N13 = N12 + NEQ
      N13 = N13 + MOD((N13-1),IPR)
      NE = N13 + NEQ*IPR
      CALL SETMEM(NE)
      CALL PZERO(M(N13),NEQ*IPR)
      GO TO 1
C.... CALL MACRO SOLUTION MODULE FOR ESTABLISHING SOLUTION ALGORITHM
200   CALL PMACR(M(N01),M(N0),M(N1),M(N2),M(N3),M(N4),M(N5),M(N6),M(N7)
     1  ,M(N8),M(N9),M(N10),M(N11),M(N12),M(N13),M,M(NE),NDF,NDM,NEN1
     2  ,NST,NE)
      GO TO 1
C.... INPUT/OUTPUT FORMATS
1000  FORMAT(20A4)
1001  FORMAT(16I5)
2000  FORMAT(1H1,20A4//5X,30HNUMBER OF NODAL POINTS        =,I6/5X,30HNUM
```

```
     1BER CF ELEMENTS             =,I6/5X,30HNUMBER OF MATERIAL SETS
     2=,I6/5X,30HDIMENSION OF COORDINATE SPACE=,I6/5X,30HDEGREES OF FREE
     3DOM/NODE        =,I6/5X,30HNODES PER ELEMENT (MAXIMUM)  =,I6/5X,30HE
     4XTRA D.O.F. TO ELEMENT        =,I6)
      END
      BLOCK DATA
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /LABEL/ PDIS(6),A(6),BC(2),DI(6),CD(3),TE(3),FD(3)
      DATA A/2H,1,2H,2,2H,3,2H,4,2H,5,2H,6/,CD/4H COO,4HRDIN,4HATES/
      DATA TE/4H TEM,4HPERA,4HTURE/,FD/4H FOR,4HCE/D,4HISPL/
      DATA PDIS/4H(I10,2H, ,4HF13.,4H4,   ,4H6E13,4H.5) /
      DATA BC/4H B.C,2H. /,DI/4H DIS,2HPL,4H VEL,2HOC,4H ACC,2HEL/
      DATA O/1HC/,IPR/2/
      END


      SUBROUTINE GENVEC(NDM,X,CD,PRT,ERR,PRTZ)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... GENERATE REAL DATA ARRAYS BY LINEAR INTERPOLATION
C
      LOGICAL PRT,ERR,PCOMP,PRTZ
      COMMON /CDATA/O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION X(NDM,1),XL(7),CD(3)
      DATA BL/4HBLAN/
      MCT = 0
      N = 0
      NG = 0
102   L = N
      LG = NG
      READ(5,1000) N,NG,XL
      IF(N.LE.0.OR.N.GT.NUMNP) GO TO 109
      DO 103 I = 1,NDM
103   X(I,N) = XL(I)
      IF(LG) 104,102,104
104   LG = ISIGN(LG,N-L)
      LI = (IABS(N-L+LG)-1)/IABS(LG)
      DO 105 I = 1,NDM
105   XL(I) = (X(I,N)-X(I,L))/LI
106   L = L + LG
      IF((N-L)*LG.LE.0) GO TO 102
      IF(L.LE.0.OR.L.GT.NUMNP) GO TO 108
      DO 107 I = 1,NDM
107   X(I,L) = X(I,L-LG) + XL(I)
      GO TO 106
108   WRITE(6,3000) L,(CD(I),I=1,3)
      ERR = .TRUE.
```

```
      GO TO 102
109   IF(.NOT.PRT) RETURN
      DO 113 J = 1,NUMNP
      IF(PRTZ) GO TO 111
      DO 110 L = 1,NDM
      IF(X(L,J).NE.0.0D0) GO TO 111
110   CONTINUE
      GO TO 113
111   MCT = MCT - 1
      IF(MCT.GT.0) GO TO 112
      MCT = 50
       WRITE(6,2000) O,HEAD,(CD(L),L=1,3),(L,CD(1),CD(2),L=1,NDM)
112   IF(PCOMP(X(1,J),BL)) WRITE(6,2008) N
      IF(.NOT.PCOMP(X(1,J),BL)) WRITE(6,2009) J,(X(L,J),L=1,NDM)
113   CONTINUE
      RETURN
1000  FORMAT(2I5,7F10.0)
2000  FORMAT(A1,20A4//5X,5HNCDAL,3A4//6X,4HNODE,9(I7,A4,A2))
2008  FORMAT(I10,32H HAS NOT BEEN INPUT OR GRNERATED)
2009  FORMAT(I10,9F13.4)
3000  FORMAT(5X,43H**FATAL ERROR 02** ATTEMPT TO GENERATE NODE,I5,3H IN
     1  ,3A4)
      END


      SUBROUTINE GENELM(ICL,IX,NEN1,PRT,ERR)
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL PRT,ERR
      COMMON /CDATA/O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION ICL(1),IX(NEN1,1)
C
C.... READ AND/OR GENERATE ELEMENT CONNECTIONS
C
      L = 0
      DO 206 I = 1,NUMEL,50
      IF(PRT) WRITE(6,2001) O,HEAD,(K,K=1,NEN)
      J = MIN0(NUMEL,I+49)
      DO 206 N = I,J
      IF(L-N) 200,202,203
200   READ(5,1001) L,LK,(ICL(K),K=1,NEN),LX
      IF(L.EQ.0) L = NUMEL+1
      IF(LX.EQ.0) LX=1
      IF(L-N) 201,202,203
201   WRITE(6,3001) L,N
      ERR = .TRUE.
      GO TO 206
202   NX = LX
      DO 207 K = 1,NEN
```

```
      IF(ICL(K).GT.NUMNP.OR.ICL(K).LT.0) GO TO 208
207   IX(K,L) = IDL(K)
      IX(NEN1,L) = LK
      GO TO 205
203   IX(NEN1,N) = IX(NEN1,N-1)
      DO 204 K = 1,NEN
      IX(K,N) = IX(K,N-1) + NX
      IF(IX(K,N-1).EQ.0) IX(K,N) = 0
      IF(IX(K,N).GT.NUMNP.OR.IX(K,N).LT.0) GO TO 208
204   CONTINUE
205   IF(PRT) WRITE(6,2002) N,IX(NEN1,N),(IX(K,N),K=1,NEN)
      GO TO 206
208   WRITE(6,3002) N
      ERR = .TRUE.
206   CONTINUE
      RETURN
1001  FORMAT(16I5)
2001  FORMAT(A1,20A4//5X,8HELEMENTS//3X,7HELEMENT,2X,8HMATERIAL,
     1   14(I3,5H NODE)/(20X,14(I3,5H NODE)))
2002  FORMAT(2I10,14I8/(20X,14I8))
3001  FORMAT(5X,20H**ERROR 03** ELEMENT,I5,22H APPEARS AFTER ELEMENT,I5)
3002  FORMAT(5X,20H**ERROR 04** ELEMENT,I5,18H HAS ILLEGAL NODES)
      END


      SUBROUTINE GENBCC(IDL,ID,NDF,PRT,III)
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL PRT
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /LABEL/ PDIS(6),A(6),BC(2),DI(6),CD(3),TE(3),FD(3)
      DIMENSION IDL(6),ID(NDF,1)
C
C.... READ AND/OR GENERATE RESTRAINT CODES
C
      IF(PRT) WRITE(6,2000) O,HEAD,(I,BC,I=1,NDF)
      III = 1
      N = 0
      NG = 0
402   L = N
      LG = NG
      READ(5,1001) N,NG,IDL
      IF(N.LE.0.OR.N.GT.NUMNP) GO TO 60
      DO 51 I = 1,NDF
      ID(I,N) = IDL(I)
      IF(L.NE.N.AND.IDL(I).EQ.0.AND.ID(I,L).LT.0) ID(I,N) = -1
51    CONTINUE
      LG = ISIGN(LG,N-L)
52    L = L + LG
```

```
      IF((N-L)*LG.LE.0) GO TO 402
      DO 53 I = 1,NDF
53    IF(ID(I,L-LG).LT.0) ID(I,L) = -1
      GO TO 52
60    DO 58 N = 1,NUMNP
      DO 56 I = 1,NDF
      IF(ID(I,N).NE.0) GO TO 57
56    CONTINUE
      GO TO 58
57    IF(PRT) WRITE(6,2001) N,(ID(I,N),I=1,NDF)
58    CONTINUE
      RETURN
1001  FORMAT(16I5)
2000  FORMAT(A1,20A4//5X,17HNODAL B.C.          //6X,4HNODE,9(I7,A4,A2)/1X)
2001  FORMAT(I10,9I13)
      END


      SUBROUTINE PMESH(IDL,IE,D,ID,X,IX,F,T,NDF,NDM,NEN1,III)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... DATA INPUT ROUTINE FOR MESH DESCRIPTION
C
      LOGICAL PRT,ERR,PCOMP
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /ELDATA/ DM,N,MA,MCT,IEL,NEL
      COMMON /LABEL/ PDIS(6),A(6),BC(2),DI(6),CD(3),TE(3),FD(3)
      DIMENSION IE(7,1),D(10,1),ID(NDF,1),X(NDM,1),IX(NEN1,1),XHED(7)
     1   ,IDL(6),F(NDF,1),T(1),WD(12),VA(2)
      DATA WD/4HCOOR,4HELEM,4HMATE,4HBOUN,4HFORC,4HTEMP,4HEND ,4HPRIN,
     1          4HNOPR,4HPAGE,4HFOLA,4HSPHE/
      DATA BL/4HBLAN/,VA/4H VAL,2HUE/,LIST/12/
C.... INITIALIZE ARRAYS
      ERR = .FALSE.
      IF(III.LT.0) GO TO 10
      PRT = .TRUE.
      DO 101 N = 1,NUMNP
      DO 100 I = 1,NDF
      ID(I,N) = 0
100   F(I,N) = 0.0D0
      X(1,N) = BL
101   T(N) = 0.0D0
10    READ(5,1000) CC
      DO 20 I = 1,LIST
      IF(PCOMP(CC,WD(I))) GO TO 30
20    CONTINUE
      GO TO 10
30    GO TO (1,2,3,4,5,6,7,8,9,11,12,13),I
```

```
C.... NODAL COORDINATE DATA INPUT
1     CALL GENVEC(NDM,X,CD,PRT,ERR,.TRUE.)
      GO TO 10
C.... ELEMENT DATA INPUT
2     CALL GENELM(IDL,IX,NEN1,PRT,ERR)
      GO TO 10
C.... MATERIAL DATA INPUT
3     WRITE(6,2004) O,HEAD
      DO 304 N = 1,NUMMAT
      READ(5,1002) MA,IEL,IDL,XHED
      IF(MA.EQ.0) GO TO 10
      DO 300 I = 1,NDF
300   IE(I,MA) = IDL(I)
      DO 301 I = 1,NDF
      IF(IDL(I).NE.0) GO TO 303
301   CONTINUE
C.... ALL ZERO RESET TO DEFAULT VALUES
      DO 302 I = 1,NDF
302   IE(I,MA) = I
303   IE(7,MA) = IEL
      WRITE(6,2003) MA,IEL,XHED,(I,IE(I,MA),I=1,NDF)
304   CALL ELMLIB(D(1,MA),IDL,X,IX,IDL,IDL,IDL,NDF,NDM,NDF,1)
      GO TO 10
C.... READ THE RESTRAINT CONDITIONS FOR EACH NODE
4     CALL GENBCC(IDL,ID,NDF,PRT,III)
      GO TO 10
C.... FORCE/DISPL DATA INPUT
5     CALL GENVEC(NDF,F,FD,PRT,ERR,.FALSE.)
      GO TO 10
C.... TEMPERATURE DATA INPUT
6     CALL GENVEC(1,T,TE,PRT,ERR,.FALSE.)
      GO TO 10
7     IF(ERR) STOP
      RETURN
8     PRT = .TRUE.
      GO TO 10
9     PRT = .FALSE.
      GO TO 10
11    READ(5,1000) O
      GO TO 10
C.... CONVERT POLAR COORDINATES TO CARTESIAN
12    CALL POLAR (X,NDM,PRT)
      GO TO 10
C.... CONVERT SPHERICAL COORDINATES TO CARTESIAN
13    CALL SPHERE (X,NDM,PRT)
      GO TO 10
1000  FORMAT(A4,75X,A1)
1002  FORMAT(I5,4X,I1,6I2,7A4)
```

```
2003  FORMAT(/5X,12HMATERIAL SET,I3,17H FOR ELEMENT TYPE,I2,5X,7A4//
     1   10X,49HDEGREE OF FREEDOM ASSIGNMENTS     LOCAL     GLOBAL /
     2   42X, 6HNUMBER, 4X, 6HNUMBER/(36X,2I10))
2004  FORMAT(A1,20A4//5X,19HMATERIAL PROPERTIES)
      END


      SUBROUTINE POLAR (X,NDM,PRT)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CONVERT POLAR TO CARTESIAN COORDINATES
C
      LOGICAL PRT
      DIMENSION X(NDM,1)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      IF(NDM.EQ.1) RETURN
      MCT = 0
      TH = DATAN(1.0D0)/45.0D0
100   READ(5,1000) NI,NE,INC,XO,YO
      IF(NI.LE.0) RETURN
      IF(NI.GT.NUMNP.OR.NE.GT.NUMNP) GO TO 300
      INC = ISIGN(MAX0(IABS(INC),1),NE-NI)
      IF(NE.EQ.0) NE = NI
      N = NI
200   R = X(1,N)
      X(1,N) = XO + R*DCOS(X(2,N)*TH)
      X(2,N) = YO + R*DSIN(X(2,N)*TH)
      MCT = MCT - 1
      IF(MCT.GT.0) GO TO 250
      IF(PRT) WRITE(6,2000) O,HEAD,XO,YO,(I,I=1,NDM)
      MCT = 50
250   IF(PRT) WRITE(6,2001) N,(X(I,N),I=1,NDM)
      N = N + INC
      IF((NE-N)*INC.GE.0) GO TO 200
      IF(MOD(NE-NI,INC).EQ.0) GO TO 100
      NI = NE
      N = NE
      GO TO 200
C
C.... ERROR
300   WRITE(6,3000) NI,NE
      STOP
C
C.... FORMATS
1000  FORMAT(3I5,5X,2F10.0)
2000  FORMAT(A1,20A4//5X,58HCARTESIAN COORDINATES COMPUTED FROM POLAR IN
     1PUT WITH XO = ,E12.4,6H YO = ,E12.4
     2 //6X,4HNODE,6(I7,6H-COORD))
```

```
2001  FORMAT(I10,6F13.4)
3000  FORMAT(5X,49H**FATAL ERROR 16** ATTEMPT TO CONVERT NODES NI =   ,
     1   I6,5H TO NE = ,I6)
      END

      SUBROUTINE SETMEM(J)
C
C.... MONITOR AVAILABLE MEMORY IN BLANK COMMON
C
      COMMON M(1)
      COMMON /PSIZE/ MAX
      K = J
      IF(K.LE.MAX) RETURN
      WRITE(6,1000) K,MAX
      STOP
1000  FORMAT(5X,49H**ERROR 01** INSUFFICIENT STORAGE IN BLANK COMMON/
     1   17X,11HREQUIRED  =,I8/17X,11HAVAILABLE =,I8/)
      END


      SUBROUTINE SPHERE (X,NDM,PRT)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CONVERT SPHERICAL TO CARTESIAN COORDINATES
C
      LOGICAL PRT
      DIMENSION X(NDM,1)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      IF(NDM.LT.3) RETURN
      MCT = 0
      TH = DATAN(1.0D0)/45.0D0
100   READ(5,1000) NI,NE,INC,XO,YO,ZO
      IF(NI.LE.0) RETURN
      IF(NI.GT.NUMNP.OR.NE.GT.NUMNP) GO TO 300
      INC = ISIGN(MAXO(IABS(INC),1),NE-NI)
      IF(NE.EQ.0) NE = NI
      N = NI
200   R = X(1,N)
      ANGX = X(2,N)*TH
      ANGZ = X(3,N)*TH
      X(1,N) = XO + R*DSIN(ANGZ)*DCOS(ANGX)
      X(2,N) = YO + R*DSIN(ANGZ)*DSIN(ANGX)
      X(3,N) = ZO + DCOS(ANGZ)
      MCT = MCT - 1
      IF(MCT.GT.0) GO TO 250
      IF(PRT) WRITE(6,2000) O,HEAD,XO,YO,ZO,(I,I=1,NDM)
      MCT = 50
```

```
250   IF(PRT) WRITE(6,2001) N,(X(I,N),I=1,NCM)
      N = N + INC
      IF((NE-N)*INC.GE.0) GC TC 200
      IF(MOD(NE-NI,INC).EQ.0) GO TO 100
      NI = NE
      N = NE
      GO TO 200
C
C.... ERRCR
300   WRITE(6,3000) NI,NE
      STCP
C
C.... FORMATS
1000  FORMAT(3I5,5X,3F10.0)
2000  FORMAT(A1,20A4//5X,62HCARTESIAN COORDINATES COMPUTED FROM SPHERICA
     1L INPUT WITH XO = ,E12.4,6H YO = ,E12.4,6H ZO = ,E12.4
     2 //6X,4HNODE,6(I7,6H-COORD))
2001  FORMAT(I10,6F13.4)
3000  FORMAT(5X,49H**FATAL ERRCR 16** ATTEMPT TO CONVERT NODES NI =  ,
     1    I6,5H TC NE = ,I6)
      END
```

```
      SUBROUTINE PMACIO(CT,WD,ENDM,ERR,IPR,LL,NE)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... MACRO PROGRAM INPUT/OUTPUT ROUTINE
C
      LOGICAL ERR,PCOMP
      DIMENSION CT(4,1),WD(1)
C.... READ MACRO CARDS
      LL = 1
      LMAX = 16
      CALL SETMEM(NE+LMAX*4*IPR)
      CT(1,1) = WD(7)
      CT(3,1) = 1.0
100   LL = LL + 1
      IF(LL.LT.LMAX) GO TO 110
      LMAX = LMAX + 16
      CALL SETMEM(NE+LMAX*4*IPR)
110   READ(5,1000) (CT(J,LL),J=1,4)
      WRITE(6,2000) (CT(J,LL),J=1,4)
      IF(.NOT.PCOMP(CT(1,LL),ENDM)) GO TO 100
200   CT(1,LL) = WD(8)
C.... SET LOOP MARKERS
      NE = NE + LMAX*4*IPR
      LX = LL-1
      DO 230 L = 1,LX
      IF(.NOT.PCOMP(CT(1,L),WD(7))) GO TO 230
      J = 1
      K = L+ 1
      DO 210 I = K,LL
      IF(PCOMP(CT(1,I),WD(7))) J = J + 1
      IF(J.GT.9) GO TO 401
      IF(PCOMP(CT(1,I),WD(8))) J = J - 1
      IF(J.EQ.0) GO TO 220
210   CONTINUE
      GO TO 400
220   CT(4,I) = L
      CT(4,L) = I
230   CONTINUE
      J = 0
      DO 240 L = 1,LL
      IF(PCOMP(CT(1,L),WD(7))) J = J+1
240   IF(PCOMP(CT(1,L),WD(8))) J = J-1
      IF(J.EQ.0) RETURN
400   WRITE(6,4000)
      ERR = .TRUE.
      RETURN
401   WRITE(6,4001)
      ERR = .TRUE.
```

```
      RETURN
 1000 FORMAT(A4,1X,A4,1X,2F10.0)
 2000 FORMAT(10X,A4,1X,A4,1X,2G15.5)
 4000 FORMAT(5X,46H**FATAL ERROR 10** UNBALANCED LOOP/NEXT MACROS )
 4001 FORMAT(5X,45H**FATAL ERROR 11** LOOPS NESTED DEEPER THAN 8)
      END


      SUBROUTINE PMACR (UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,B,DR,CT
     1,NDF,NDM,NEN1,NST,NEND)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... MACRO INSTRUCTION SUBPROGRAM
C
C.... CONTROLS PROBLEM SOLUTION AND OUTPUT ALGORITHMS BY
C.... ORDER OF SPECIFYING MACRO COMMANDS IN ARRAY WD.
C
      LOGICAL AFR,BFR,CFR,AFL,BFL,CFL,DFL,EFL,FFL,GFL,TFL,PCOMP
      LOGICAL ERR,EER
      COMMON M(1)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /ENGYS/ AENGY
      COMMON /LABEL/ PDIS(6),Z(6),BC(2),DI(6),CD(3),TE(3),FD(3)
      COMMON /MDATA/ NO1,NC,N1,N2,N3,N4,N5,N6,N7,N8,N9,N10,N11,N12,N13
      COMMON /PRLOD/ PROP
      COMMON /PSIZE/ MAX
      COMMON /TDATA/ TIME,DT,C1,C2,C3,C4,C5
      DIMENSION WD(25),CT(4,1),CTL(4),LVS(9),LVE(9),JDIAG(1),
     1   UL(1),XL(1),TL(1),LD(1),P(1),S(1),IE(7,1),D(1),ID(1),X(1),
     2   IX(1),F(1),T(1),B(1),DR(1)
      DATA WD/4HTOL ,4HDT  ,4HSTRE,4HDISP,4HTANG,4HFORM,4HLOOP,4HNEXT,
     1        4HPROP,4HDATA,4HTIME,4HCONV,4HSCLV,4HLMAS,4HCMAS,4HMESH,
     2        4HEIGE,4HEXCD,4HUTAN,4HREAC                  ,4HRITE,
     3                      4HCHEC,                         4HODE1,4HUSER,
     4        4H*TIM/
      DATA NWD/25/,ENDM/4HEND /,NV/1/,NC/1/,NA/1/
      INTEGER CLOCK(6)
C.... SET INITIAL VALUES OF PARAMETERS
      CT = 0.0D0
      PROP = 1.0D0
      RNMAX = 0.0D0
      TIME = 0.0D0
      TOL = 1.0D-9
      UN = 0.0D0
      AFL = .TRUE.
      AFR = .FALSE.
      BFL = .TRUE.
      BFR = .FALSE.
```

```
      CFL = .TRUE.                                                       ...00546
      CFR = .FALSE.                                                      ...00547
      DFL = .TRUE.                                                       ...00548
      EFL = .TRUE.                                                       ...00549
      FFL = .FALSE.                                                      ...00550
      GFL = .TRUE.                                                       ...00551
      TFL = .FALSE.                                                      ...00552
      NE = NENC                                                          ...00553
      NNEG = NDF*NUMNP                                                   ...00554
      NPLD = 0                                                           ...00555
      WRITE(6,2001) O,HEAD                                               ...00556
C.... READ MACRO CARDS                                                    ...00557
      ERR = .FALSE.                                                      ...00558
      CALL PMACIO(CT,WD,ENDM,ERR,IPR,LL,NE)                              ...00559
      IF(ERR) RETURN                                                     ...00560
C.... EXECUTE MACRO INSTRUCTION PROGRAM                                   ...00561
      LV = 0                                                             ...00562
      L = 1                                                              ...00563
299   I = L - 1                                                          ...00564
      DO 300 J = 1,NWD                                                   ...00565
      IF(PCOMP(CT(1,L),WD(J))) GO TO 310                                 ...00566
300   CONTINUE                                                           ...00567
      WRITE(6,3000) I                                                    ...00568
      GO TO 330                                                          ...00569
310   CONTINUE                                                           ...00570
      IF(L.EQ.1.OR.L.EQ.LL) GO TO 311                                    ...00571
      WRITE(6,2010) I,(CT(K,L),K=1,4)                                    ...00572
      IF(.NOT.TFL) GO TO 311                                             ...00573
      CALL SETIME                                                        ...00574
      WCTM = ITIME(0)*0.01D0                                             ...00575
311   LLL = L                                                            ...00576
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,         ...00577
     123,        27,          31,32,33),J                                ...00578
C.... SET SOLUTION TOLERANCE                                              ...00579
1     TOL = CT(3,L)                                                      ...00580
      GO TO 330                                                          ...00581
C.... SET TIME INCREMENT                                                  ...00582
2     DT = CT(3,L)                                                       ...00583
      GO TO 330                                                          ...00584
C.... PRINT STRESS VALUES                                                 ...00585
3     LX = LVE(LV)                                                       ...00586
      IF(DMOD(CT(3,LX),DMAX1(CT(3,L),1.0D0)).EQ.0.0D0)                   ...00587
     1 CALL PFORM(UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,DR,DR,DR,       ...00588
     2   NDF,NDM,NEN1,NST,4,B,M(NV),.FALSE.,.FALSE.,.FALSE.,.FALSE.,1)   ...00589
      GO TO 330                                                          ...00590
C.... PRINT DISPLACEMENTS                                                 ...00591
4     LX = LVE(LV)                                                       ...00592
      IF(DMOD(CT(3,LX),DMAX1(CT(3,L),1.0D0)).NE.0.0D0) GO TO 330         ...00593
```

```
      WRITE(6,2003) O,HEAD,TIME,PROP
      CALL PRTDIS(ID,X,B,F,NDM,NDF)
      GO TO 330
C.... FORM TANGENT STIFFNESS
19    IF(CFL) CALL PSETM(NC,NE,JDIAG(NEQ)*IPR,CFL)
      CALL PZERO(M(NC),JDIAG(NEQ)*IPR)
      CFR = .TRUE.
5     IF(J.EQ.5) CFR = .FALSE.
      IF(GFL) CALL PSETM(NA,NE,JDIAG(NEQ)*IPR,GFL)
      IF(NFLD.GT.0) PROP = PROPLD(TIME,0)
      CALL PZERO(M(NA),JDIAG(NEQ)*IPR)
      CALL PFORM(UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,DR,M(NA),M(NC),
     2   NDF,NDM,NEN1,NST,3,B,M(NV),.TRUE.,.FALSE.,CFR,.FALSE.,1)
      AFR = .TRUE.
      GO TO 330
C.... FORM OUT OF BALANCE FORCE FOR TIME STEP/ITERATION
6     IF(NFLD.GT.0) PROP = PROPLD(TIME,0)
      CALL PLOAD(ID,F,DR,NNEQ,PROP)
      CALL PFORM(UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,DR,M(NA),M(NC),
     2   NDF,NDM,NEN1,NST,6,B,M(NV),.FALSE.,.TRUE.,.FALSE.,.FALSE.,1)
      BFR = .TRUE.
      RN = DSQRT(DOT(DR,DR,NEQ))
      RNMAX = DMAX1(RNMAX,RN)
      WRITE(6,2005) RNMAX,RN,TOL
      IF(RN.GE.RNMAX*TOL) GO TO 330
      LX = LVE(LV)
      LO = LVS(LV)
      CT(3,LX) = CT(3,LO)
      L = LX - 1
      GO TO 330
C.... SET LOOP START INDICATORS
7     LV = LV + 1
      LX = CT(4,L)
      LVS(LV) = L
      LVE(LV) = LX
      CT(3,LX) = 1.0D0
      GO TO 330
C.... LOOP TERMINATOR CONTROL
8     N = CT(4,L)
      CT(3,L) = CT(3,L) + 1.0D0
      IF(CT(3,L).GT.CT(3,N)) LV = LV -1
      IF(CT(3,L).LE.CT(3,N)) L = N
      GO TO 330
C.... INPUT PROPORTIONAL LOAD TABLE
9     NPLD = CT(3,L)
      PROP = PROPLD(0.0D0,NPLD)
      GO TO 330
C.... READ COMMAND
```

```
10     READ(5,1000) (CTL(I),I=1,4)
       IF(.NOT.FCOMP(CT(2,L),CTL(1))) GO TO 402
       IF(PCOMP(CTL(1),WD(1))) TOL = CTL(3)
       IF(PCOMP(CTL(1),WD(2))) DT = CTL(3)
       IF(PCOMP(CTL(1),WD(4))) GO TO 101
       GO TO 330
101    ISW = CT(3,L)
       IF(ISW.EQ.0) ISW = 5
       IF(ISW.NE.5) REWIND ISW
       READ(ISW) TIME,(B(I),I=1,NEQ)
       GO TO 330
C..... INCREMENT TIME
11     TIME = TIME + DT
       RNMAX = 0.0D0
       UN = 0.0D0
       GO TO 330
C..... COMPUTE CONVERGENCE TESTS
12     RN = DOT(DR,DR,NEQ)
       UN = UN + DOT(B,B,NEQ)
       UN = DMAX1(UN,RN)
       CN = DSQRT(UN)
       RN = DSQRT(RN)
       WRITE(6,2002) CN,RN,TOL
       LX = LVE(LV)
       LO = LVS(LV)
       IF(RN.LT.CN*TOL) CT(3,LX) = CT(3,LO)
       GO TO 330
C..... SOLVE THE EQUATIONS
13     IF(CFR) GO TO 131
       CALL ACTCOL(M(NA),DR,JDIAG,NEQ,AFR,BFR)
       IF(BFR) WRITE(6,2007) AENGY
       GO TO 132
131    CALL UACTCL(M(NA),M(NC),DR,JDIAG,NEQ,AFR,BFR)
132    AFR = .FALSE.
       IF(.NOT.BFR) GO TO 330
       BFR = .FALSE.
       DO 133 N = 1,NEQ
133    B(N) = B(N) + DR(N)
       GO TO 330
C..... FORM A LUMPED MASS APPROXIMATION
14     AFL = .FALSE.
       BFL = .TRUE.
       IF(BFL) CALL PSETM(NN,NE,NEQ*IPR,BFL)
139    CALL PZERO (M(NN),NEQ*IPR)
       GO TO 140
C..... FORM A CONSISTENT MASS APPROXIMATION
15     AFL = .TRUE.
       BFL = .FALSE.
```

```
      IF(DFL) CALL PSETM(NM,NE,JDIAG(NEQ)*IPR,DFL)
152   CALL PZERO(M(NM),JDIAG(NEQ)*IPR)
140   CALL PFORM(UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,M(NN),M(NM),
     1  M(NM),NDF,NDM,NEN1,NST,5,B,M(NV),AFL,BFL,.FALSE.,.FALSE.,4)
      GO TO 330
C.... CHANGE MESH DATA
16    I = -1
      CALL PMESH(LD,IE,D,ID,X,IX,F,T,NDF,NDM,NEN1,I)
      IF(I.GT.0) GO TO 404
      GO TO 330
C.... COMPUTE DOMINANT EIGENVALUE BY INVERSE ITERATION
17    J = NM
      IF(DFL) J = NN
      CALL PEIGS(M(NA),M(J),F,X,B,DR,ID,IX,JDIAG,NDF,NDM,NEN1,DFL)
      GO TO 330
18    IF(FFL) GO TO 181
C.... MACRO *EXCD* EXPLICIT INTEGRAIION OF EQUATIONS OF MOTION
      NQ = NE
      NQ = NQ + MOD((NQ-1),IPR)
      NV = NC + NEQ*IPR
      NR = NV + NDF*NUMNP*IPR
      NE = NR + NEQ*IPR
      CALL SETMEM(NE+1)
      CALL PZERO(M(NQ),NE-NC)
      FFL = .TRUE.
      GO TO 330
181   IF(.NOT.BFR.OR.BFL) GO TO 403
      CALL ELPDAT(DR,B,M(NC),M(NR),M(NN),DT,NEQ)
      GO TO 330
C.... COMPUTE REACTIONS AND PRINT
20    CALL PZERO(DR,NNEQ*IPR)
      CALL PFORM(UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,DR,DR,DR,
     1    NDF,NDM,NEN1,NST,6,B,M(NV),.FALSE.,.TRUE.,.FALSE.,.TRUE.,1)
      CALL PRTREA(DR,NDF)
      GO TO 330
C.... PRINT ARRAYS AS OUTPUT
23    IF(PCOMP(CT(2,L),WD(5)).OR.PCOMP(CT(2,L),WD(15)))
     1  WRITE (6,2006) (N,JDIAG(N),N=1,NEQ)
      IF(PCOMP(CT(2,L),WD(5)).AND.AFR) CALL PRITE(M(NA),JDIAG(NEQ),
     1  9HSTIFFNESS )
      IF(PCOMP(CT(2,L),WD(6)).AND.BFR) CALL PRITE(DR,NEQ,
     1  9H   FORCE )
      IF(PCOMP(CT(2,L),WD(15)).AND.AFL) CALL PRITE(M(NM),JDIAG(NEQ),
     1  15HCONSISTENT MASS  )
      IF(PCOMP(CT(2,L),WD(14)).AND.BFL) CALL PRITE(M(NN),NEQ,
     1  11HLUMPED MASS )
      GO TO 330
C.... CHECK MESH FOR ERRORS
```

```
27    CALL PFORM(UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,DR,DR,DR,
     1   NDF,NDM,NEN1,NST,2,B,M(NV),.FALSE.,.FALSE.,.FALSE.,.FALSE.,1)
      GO TO 330
C.... FIRST ORDER ODE SOLUTION
31    CALL PODE1 (CT(2,L),M(NA),M(NC),M(NM),M(NN),B,DR,JDIAG,ID,UL,
     1  (N12-N01),NDF,AFR,CFR,AFL,BFL,EER)
      IF(EER) WRITE (6,3000) I
      GO TO 330
C.... USER SUPPLIED SUBROUTINE
32    STOP
C.... SET TIME COUNTER
33    TFL = .TRUE.
      GO TO 330
330   L = L + 1
      IF (LLL.EQ.1.OR.LLL.EQ.LL) GO TO 331
      IF(.NOT.TFL) GO TO 331
      CALL GETIME(IET)
      WCTM = ITIME(0)*0.01D0 - WCTM
      CPUTM = IET * 0.000026D0
      WRITE(6,2015) I,(CT(K,LLL),K=1,4),WCTM,CPUTM
331   CONTINUE
      IF(L.LE.LL) GO TO 299
      WRITE(6,2008)
      RETURN
402   WRITE(6,4002)
      RETURN
403   WRITE(6,4003)
404   WRITE(6,4004)
      RETURN
1000  FORMAT(A4,1X,A4,1X,2F10.0)
2001  FORMAT(A1,20A4//5X,18HMACRO INSTRUCTIONS//5X,15HMACRO STATEMENT,5X
     1,10HVARIABLE 1,5X,10HVARIABLE 2)
2002  FORMAT(5X,29HDISPLACEMENT CONVERGENCE TEST/10X,7HUNMAX =,G15.5,5X,
     1   7HUN    =,G15.5,5X,7HTOL   =,G15.5)
2003  FORMAT(A1,20A4,10X,4HTIME,G13.5//5X,17HPROPORTIONAL LOAD,G13.5)
2004  FORMAT(5X,4HCN =,G12.5,5X,4HDN =,G12.5,5X,4HUN =,G12.5,5X,4HAG =
     1   ,G12.5,5X,4HAC =,G12.5)
2005  FORMAT(5X,22HFORCE CONVERGENCE TEST/10X,7HRNMAX =,G15.5,5X,
     1   7HRN    =,G15.5,5X,7HTOL   =,G15.5)
2006  FORMAT(/10X,22HDIAGONAL ADDRESS ARRAY /(1X,20I5))
2007  FORMAT(5X,18HENERGY (DR*A*DR) =   ,E18.10)
2008  FORMAT(2X,26H**END OF MACRO EXECUTION**)
2010  FORMAT(2X,19H**MACRO INSTRUCTION,I4,20H START EXECUTION**  ,2(A4,2
     1X), 6H V1 = ,G13.4, 8H , V2 = ,G13.4)
2015  FORMAT(2X,19H**MACRO INSTRUCTION,I4,13H EXECUTED**  ,2(A4,2X), 6H
     1V1 = ,G13.4, 8H , V2 = ,G13.4,/
     25X,20H   WALL CLOCK TIME = ,F7.2,7HSECONDS,
     35X,13H   CPU TIME = ,F7.2,7HSECONDS)
```

```
3000  FORMAT(5X,29H**WARNING** MACRO INSTRUCTION   ,I4,
     1   18H IS NOT EXECUTABLE  )
4002  FORMAT(5X,57H**FATAL ERROR 12** MACRO LABEL MISMATCH ON A READ COM
     1MAND)
4003  FORMAT(5X,63H**FATAL ERROR 13** MACRO EXCD MUST BE PRECEDED BY LMA
     1S AND FORM)
4004  FORMAT(5X,84H**FATAL ERROR 14** ATTEMPT TO CHANGE BOUNDARY RESTRAI
     1NT CODES DURING MACRO EXECUTION )
      END
```

```
      SUBROUTINE ACTCOL(A,B,JDIAG,NEQ,AFAC,BACK)
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL AFAC,BACK
      COMMON/ENGYS/AENGY
      DIMENSION A(1),B(1),JDIAG(1)
      DATA TOL/1.D-7/
C
C.... ACTIVE COLUMN PROFILE SYMMETRIC EQUATION SOLVER
C
C.... FACTOR A TO UT*D*U, REDUCE B
      AENGY = 0.0D0
      JR = 0
      DO 600 J = 1,NEQ
      JD = JDIAG(J)
      JH = JD - JR
      IS = J - JH + 2
      DG = A(JD)
      IF(JH-2) 550,300,100
100   IF(.NOT.AFAC) GO TO 500
      IE = J - 1
      K = JR + 2
      ID = JDIAG(IS -1)
C.... REDUCE ALL EQUATIONS EXCEPT DIAGONAL
      DO 200 I = IS,IE
      IR = ID
      ID = JDIAG(I)
      IH = MIN0(ID-IR-1,I-IS+1)
      IF(IH.GT.0) A(K) = A(K) - DOT(A(K-IH),A(ID-IH),IH)
200   K = K + 1
C.... REDUCE DIAGONAL TERM
300   IF(.NOT.AFAC) GO TO 500
      IR = JR + 1
      IE = JD - 1
      K = J - JD
      DO 400 I = IR,IE
      ID = JDIAG(K+I)
      D = A(I)
      A(I) = A(I)*A(ID)
      A(JD) = A(JD) - D*A(I)
400   CONTINUE
      IF(DG*A(JD).LT.0.0D0) WRITE(6,2000) J
      IF(DABS(A(JD)).LT.TOL*DABS(DG)) WRITE(6,2001) J
      IF(A(JD).EQ.0.0D0) WRITE (6,2002) J
C.... REDUCE RHS
500   IF(BACK) B(J) = B(J) - DOT(A(JR+1),B(IS-1),JH-1)
550   IF(A(JD).NE.0.0D0.AND.AFAC) A(JD) = 1.0D0/A(JD)
600   JR = JD
      IF(.NOT.BACK) RETURN
```

```
C.... DIVIDE BY DIAGONAL PIVOTS
      DO 700 I = 1,NEQ
      ID = JDIAG(I)
      AENGY = AENGY + B(I)*B(I)*A(ID)
700   B(I) = B(I)*A(ID)
C.... BACKSUBSTITUTE
      J = NEQ
      JD = JDIAG(J)
800   D = B(J)
      J = J - 1
      IF(J.LE.0) RETURN
      JR = JDIAG(J)
      IF(JD-JR.LE.1) GO TO 1000
      IS = J - JD + JR + 2
      K = JR - IS + 1
      DO 900 I = IS,J
900   B(I) = B(I) - A(I+K)*D
1000  JD = JR
      GO TO 800
2000  FORMAT(5X,31H**WARNING 01** SIGN OF EQUATION,I5,18H CHANGED IN ACT
     1COL)
2001  FORMAT(5X,23H**WARNING 02** EQUATION,I5,45H LOST AT LEAST 7 SIGNIF
     1ICANT DIGITS IN ACTCOL)
2002  FORMAT(5X,32H**WARNING 03** PIVOT OF EQUATION,I5,18H IS ZERO IN AC
     1TCOL)
      END


      SUBROUTINE ADDSTF(A,B,C,S,P,JDIAG,LD,NST,NEL,AFL,BFL,CFL,ITP)
C
C.... ASSEMBLE GLOBAL ARRAYS
C
      IMPLICIT REAL*8 (A-H,O-Z)
      LOGICAL AFL,BFL,CFL
      DIMENSION A(1),B(1),JDIAG(1),P(1),S(NST,1),LD(1) ,C(1)
      DO 200 J = 1,NEL
      K = IABS(LD(J))
      IF(K.EQ.0) GO TO 200
      IF(BFL) B(K) = B(K) + P(J)
      IF(.NOT.AFL.AND..NOT.CFL) GO TO 200
      L = JDIAG(K) - K
      DO 100 I = 1,NEL
      M = IABS(LD(I))
      IF(M.GT.K.OR.M.EQ.0) GO TO 100
      M = L + M
      IF(AFL) A(M) = A(M) + S(I,J)
      IF(CFL) C(M) = C(M) + S(J,I)
100   CONTINUE
```

```
200   CCNTINUE
      RETURN
      END


      FUNCTICN CCT(A,B,N)
      IMFLICIT FEAL*8 (A-H,C-Z)
C
C.... VECTCR CCT FRODUCT
C
      CIMENSICN A(1),B(1)
      CCT = 0.0C0
      DO 1CC I = 1,N
100   COT = CCT + A(I)*B(I)
      RETURN
      END
      LOGICAL FUNCTION PCCMP(A,B)
      IMPLICIT REAL*8 (A-H,C-Z)
      PCOMP = .FALSE.
C.... IT MAY BE NECESSARY TC REPLACE THE FOLLOWING ALPHANUMERIC
C.... COMPARISCN STATEMENT IF CCMPUTER PRODUCES AN OVERFLOW
      IF(A.EC.B) PCOMP = .TRUE.
      RETURN
      END


      SUBRCUTINE ELMLIB(D,U,X,IX,T,S,P,I,J,K,ISW)
      IMFLICIT REAL*8 (A-H,C-Z)
C
C.... ELEMENT LIERARY
C
      COMMON /ELCATA/ DM,N,MA,MCT,IEL,NEL
      CIMENSICN P(K),S(K,K),D(1),U(1),X(1),IX(1),T(1)
      IF(IEL.LE.C.CR.IEL.GT.5) GO TO 400
      IF(ISW.LT.3) GO TO 3C
      CO 20 L = 1,K
      P(L) = C.CD0
      CO 2C M = 1,K
20    S(L,M) = C.0C0
30    GO TC (1,2,3,4,5),IEL
1     CALL ELMT01(C,U,X,IX,T,S,P,I,J,K,ISW)
      GO TC 1C
2     CALL ELMTC2(D,U,X,IX,T,S,P,I,J,K,ISW)
      GO TC 10
3     CALL ELMTC3(D,U,X,IX,T,S,P,I,J,K,ISW)
      GO TC 1C
4     CALL ELMTC4(C,U,X,IX,T,S,P,I,J,K,ISW)
      GO TC 1C
```

```
5       CALL ELMTC5(D,U,X,IX,T,S,P,I,J,K,ISW)
10      RETURN
400     WRITE(6,4000) IEL
        STOP
4000    FORMAT(5X,39H**FATAL ERROR 04** ELEMENT CLASS NUMBER,I3,6H INPUT)
        END


        SUBROUTINE EUPDAT(DR,U,V,A,XM,DT,NEQ)
        IMPLICIT REAL*8 (A-H,O-Z)
        DIMENSION U(1),V(1),A(1),DR(1),XM(1)
C.....  UPDATE SOLUTION USING EXPLICIT CENTRAL DIFFERENCES
        DATA DTHP/0.0D0/
        DTH = DT/2.0D0
        DTAV = DTH + DTHP
        DTHP = DTH
        DO 100 N = 1,NEQ
        A(N) = DR(N)/XM(N)
        V(N) = V(N) + DTAV*A(N)
100     U(N) = U(N) + DT*V(N)
        RETURN
        END


        SUBROUTINE NORM(X,Y,N)
        IMPLICIT REAL*8 (A-H,O-Z)
C
C.....  NORMALIZE Y TO UNIT VECTOR X
C
        DIMENSION X(1),Y(1)
        SCALE = DSQRT(DOT(Y,Y,N))
        DO 100 I = 1,N
100     X(I) = Y(I)/SCALE
        RETURN
        END
 N

        SUBROUTINE PADDM(A,B,C,NN)
        IMPLICIT REAL*8 (A-H,O-Z)
C
C.....  ADD LINEAR COMBINATION OF TWO ARRAYS TOGETHER
C
        DIMENSION A(1),B(1)
        DO 100 N=1,NN
100     A(N) = A(N) + C*B(N)
        RETURN
        END
```

```
      SUBROUTINE PADDV(A,B,ID,C,NN)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),ID(1)
      DO 100 N=1,NN
      K = ID(N)
      IF(K.GT.0) A(N) = A(N) + C*B(K)
100   CONTINUE
      RETURN
      END


      SUBROUTINE PADDD(A,B,JD,C,NN)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... ADD LINEAR COMBINATION OF DIAGONAL ARRAY TO DIAGONAL OF AN ARRAY
C
      DIMENSION A(1),B(1),JD(1)
      DO 100 N=1,NN
      J = JD(N)
100   A(J) = A(J) + C*B(N)
      RETURN
      END


      SUBROUTINE PSCAL (A,C,NN)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... SCALE AN ARRAY BY A CONSTANT
C
      DIMENSION A(1)
      DO 100 N=1,NN
100   A(N) = C*A(N)
      RETURN
      END


      SUBROUTINE PRODIA (A,B,C,NEQ)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... ROUTINE TO FORM C = C + A*B WHERE A IS A DIAGONAL MATRIX,
C.... B AND C ARE VECTORS
C
      DIMENSION A(1),B(1),C(1)
      DO 10 N=1,NEQ
10    C(N) = C(N) + A(N)*B(N)
      RETURN
      END
```

```
      SUBROUTINE PROMUL(A,B,C,JDIAG,NEQ)
      IMPLICIT REAL*8 (A-H,O-Z)
      DIMENSION A(1),B(1),C(1),JDIAG(1)
C
C.... ROUTINE TO FORM C = C + A*B WHERE A IS A SYMMETRIC SQUARE MATRIX
C.... STORED IN PROFILE FORM,B,C ARE VECTORS, AND JDIAG LOCATES THE
C.... DIAGONALS IN A
C
      JS = 1
      DO 200 J= 1,NEQ
      JD = JDIAG(J)
      IF(JS.GT.JD) GO TO 200
      BJ = B(J)
      AB = A(JD)*BJ
      IF(JS.EQ.JD) GO TO 150
      JB = J - JD
      JE = JD - 1
      DO 100 JJ = JS,JE
      AB = AB + A(JJ)*B(JJ+JB)
100   C(JJ+JB) = C(JJ+JB) + A(JJ)*BJ
150   C(J) = C(J) + AB
200   JS = JD + 1
      RETURN
      END


      FUNCTION PROPLD(T,J)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... PROPORTIONAL LOAD TABLE (ONE LOAD CARD ONLY)
C
      DIMENSION A(5)
      IF(J.GT.0) GO TO 200
C.... COMPUTE VALUE AT TIME T
      PROPLD = 0.0D0
      IF(T.LT.TMIN.OR.T.GT.TMAX) RETURN
      L = MAX0(L,1)
      PROPLD = A(1) + A(2)*T + A(3)*(DSIN(A(4)*T+A(5)))**L
      RETURN
C.... INPUT TABLE OF PROPORTIONAL LOADS
200   I = 1
      READ(5,1000) K,L,TMIN,TMAX,A
      WRITE(6,2000) I,K,L,TMIN,TMAX,A
      RETURN
1000  FORMAT(2I5,7F10.0)
2000  FORMAT(5X,23HPROPORTIONAL LOAD TABLE//24H  NUMBER   TYPE    EXP.,
```

```
     1 14H   MINIMUM TIME,15H    MAXIMUM TIME, 5X,2HA1,13X,2HA2,13X,2HA3,
     2   13X,2HA4,13X,2HA5/(3I8,7G15.5))
      END

      SUBROUTINE PRTDIS(ID,X,B,F,NDM,NDF)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... OUTPUT NODAL VALUES
C
      LOGICAL PCOMP
      COMMON/PRLOD/ PROP
      COMMON/CDATA/O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON/LABEL/ PDIS(6),A(6),BC(2),DI(6),CD(3),TE(3),FD(3)
      COMMON/TDATA/ TIME,DT,C1,C2,C3,C4,C5
      DIMENSION X(NDM,1),B(1),UL(6),ID(NDF,1),F(NDF,1)
      DATA BL/4HBLAN/
      DO 102 II = 1,NUMNP,50
      WRITE(6,2000) O,HEAD,TIME,(I,CD(1),CD(2),I=1,NDM),(I,DI(1)
     1   ,DI(2),I=1,NDF)
      JJ = MIN0(NUMNP,II+49)
      DO 102 N = II,JJ
      IF(PCOMP(X(1,N),BL)) GO TO 101
      DO 100 I = 1,NDF
      UL(I) = F(I,N)*PROP
      K = IABS(ID(I,N))
100   IF(K.GT.0) UL(I) = B(K)
      WRITE(6,PDIS) N,(X(I,N),I=1,NDM),(UL(I),I=1,NDF)
101   CONTINUE
102   CONTINUE
      RETURN
2000  FORMAT(A1,20A4//5X,19HNODAL DISPLACEMENTS,5X,4HTIME,E13.5//
     1   6X,4HNODE,9(I7,A4,A2))
      END

      SUBROUTINE PRTIE (A,NN,WD)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... OUTPUT ARRAY
C
      DIMENSION A(NN),WD(2)
      WRITE(6,2000) WD
      DO 100 I=1,NN,7
      K = MIN0(NN,I+6)
100   WRITE(6,2001) (J,A(J),J=I,K)
      RETURN
2000  FORMAT(/10X,2A8,6H ARRAY /)
```

```
2001  FORMAT(1X,7(I5,1PE12.4))
      END

      SUBROUTINE PROFIL (JDIAG,ID,IX,NDF,NEN1,NAD)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... COMPUTE PROFILE OF GLOBAL ARRAYS
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION JDIAG(1),ID(NDF,1),IX(NEN1,1)
C.... SET UP THE EQUATION NUMBERS
      NEQ= 0
      DO 50 N = 1,NUMNP
      DO 40 I = 1,NDF
      J=ID(I,N)
      IF(J) 30,20,30
20    NEQ = NEQ + 1
      ID(I,N) = NEQ
      JDIAG(NEQ) = 0
      GO TO 40
30    ID(I,N) = 0
40    CONTINUE
50    CONTINUE
C.... COMPUTE COLUMN HEIGHTS
      DO 500 N = 1,NUMEL
      DO 400 I = 1,NEN
      II = IABS(IX(I,N))
      IF(II.EQ.0) GO TO 400
      DO 300 K = 1,NDF
      KK = ID(K,II)
      IF(KK.EQ.0) GO TO 300
      DO 200 J = I,NEN
      JJ = IABS(IX(J,N))
      IF(JJ.EQ.0) GO TO 200
      DO 100 L = 1,NDF
      LL = ID(L,JJ)
      IF(LL.EQ.0) GO TO 100
      M = MAX0(KK,LL)
      JDIAG(M) = MAX0(JDIAG(M),IABS(KK-LL))
100   CONTINUE
200   CONTINUE
300   CONTINUE
400   CONTINUE
500   CONTINUE
C.... COMPUTE DIAGONAL POINTERS FOR PROFILE
      NAD = 1
      JDIAG(1) = 1
```

```
      IF(NEQ.EQ.1) RETURN
      DO 600 N = 2,NEQ
600   JDIAG(N) = JDIAG(N) + JDIAG(N-1) + 1
      NAD = JDIAG(NEQ)
      RETURN
      END


      SUBROUTINE PLOAD(ID,F,B,NN,P)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... FORM LOAD VECTOR IN COMPACT FORM
C
      DIMENSION ID(1),F(1),B(1)
      DO 100 N = 1,NN
      J = ID(N)
100   IF(J.GT.0) B(J) = F(N)*P
      RETURN
      END


      SUBROUTINE PFORM(UL,XL,TL,LD,P,S,IE,D,ID,X,IX,F,T,JDIAG,B,A,C,NDF,
     1   NDM,NEN1,NST,ISW,U,UD,AFL,BFL,CFL,DFL,ITP)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... COMPUTE ELEMENTS   ARRAYS AND ASSEMBLE GLOBAL ARRAYS
C
      LOGICAL AFL,BFL,CFL,DFL
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /ELDATA/ DM,N,MA,MCT,IEL,NEL
      COMMON /PRLOD/ PROP
      DIMENSION XL(NDM,1),LD(NDF,1),P(1),S(NST,1),IE(7,1),D(10,1),ID(NDF
     1,1),X(NDM,1),IX(NEN1,1),F(NDF,1),JDIAG(1),B(1),A(1),C(1),UL(NDF,1)
     2   ,TL(1),T(1),U(1),UD(NDF,1)
C.... LOOP ON ELEMENTS
      IEL = 0
      DO 110 N = 1,NUMEL
      MA = IX(NEN1,N)
C.... SET UP LOCAL ARRAYS
      DO 108 I = 1,NEN
      II = IABS(IX(I,N))
      IF(II.NE.0) GO TO 105
      TL(I) = 0.0D0
      DO 103 J = 1,NDM
103   XL(J,I) = 0.0D0
      DO 104 J = 1,NDF
      UL(J,I) = 0.0D0
      UL(J,I+NEN) = 0.0D0
```

```
104    LD(J,I) = 0
       GO TO 108
105    IID = II*NDF - NDF
       NEL = I
       TL(I) = T(II)
       DO 106 J = 1,NDM
106    XL(J,I) = X(J,II)
       DO 107 J = 1,NDF
       JJ = IE(J,MA)
       IF(JJ.LE.0) GO TO 107
       K = IABS(ID(JJ,II))
       UL(J,I) = F(JJ,II)*PROP
       UL(J,I+NEN) = UD(JJ,II)
       IF(K.GT.0) UL(J,I) = U(K)
       IF(DFL) K = IID + JJ
107    LD(J,I) = K
       IF(IX(I,N).GT.0) GO TO 108
       DO 109 J = 1,NDF
109    LD(J,I) = -LD(J,I)
108    CONTINUE
C....  FORM ELEMENT ARRAY
       IF(IE(7,MA).NE.IEL)  MCT = 0
       IEL = IE(7,MA)
       CALL ELMLIB(D(1,MA),UL,XL,IX(1,N),TL,S,P,NDF,NDM,NST,ISW)
C....  ADD TO TOTAL ARRAY
       IF(AFL.OR.BFL.OR.CFL) CALL ADDSTF(A,B,C,S,P,JDIAG,LD,NST,NEL*NDF,
     1   AFL,BFL,CFL,ITP)
110    CONTINUE
       END FILE ITP
       RETURN
       END


       SUBROUTINE PEIGS(A,B,F,X,Y,Z,ID,IX,JDIAG,NDF,NDM,NEN1,DFL)
       IMPLICIT REAL*8 (A-H,O-Z)
C
C....  COMPUTE DOMINANT EIGENVALUE BY INVERSE ITERATION
C
       LOGICAL DFL
       COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
       COMMON /ENGYS/ AENGY
       DIMENSION A(1),B(1),F(1),X(1),Y(1),Z(1),ID(1),IX(1),JDIAG(1)
       DATA ITS/100/,TOL/1.0D-9/
C....  GET START VECTOR FROM DIAGONAL OF MASS MATRIX
       DO 100 I = 1,NEQ
       J = JDIAG(I)
       IF(DFL) J = I
100    Y(I) = B(J)
```

```
      EIGP = 0.0D0
      CALL ACTCOL(A,Z,JDIAG,NEQ,.TRUE.,.FALSE.)
      DO 200 I = 1,ITS
      CALL PZERO(Z,NEQ*IPR)
      CALL PROMUL(B,Y,Z,JDIAG,NEQ)
C.... RAYLEIGH QUOTIENT
      EIG = AENGY/DOT(Y,Z,NEQ)
      IF(DABS(EIG-EIGP).LT.TOL*DABS(EIG)) GO TO 300
      CALL NORM(Y,Z,NEQ)
      EIGP = EIG
C.... INVERSE ITERATION
200   CALL ACTCOL(A,Y,JDIAG,NEQ,.FALSE.,.TRUE.)
      WRITE(6,2001) ITS
      RETURN
300   WRITE(6,2000) O,HEAD,EIG,I
      CALL NORM(Z,Y,NEQ)
      CALL PRTDIS(ID,X,Z,F,NDM,NDF)
      RETURN
2000  FORMAT(A1,20A4//5X,14HEIGENVALUE =   ,G13.4/5X,14HITERATIONS =   ,
     1   I9/)
2001  FORMAT(5X,57H**FATAL ERROR 09** NO CONVERGENCE IN EIGENVALUES, ITS
     1 =   ,I5)
      END


      SUBROUTINE PRTREA(R,NDF)
      IMPLICIT REAL*8 (A-H,O-Z)
C.... PRINT NODAL REACTIONS
C
      DIMENSION R(NDF,1),RSUM(6),ASUM(6)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DO 50 K=1,NDF
      RSUM(K) = 0.0D0
50    ASUM(K) = 0.0D0
      DO 100 N=1,NUMNP,50
      J= MINO(NUMNP,N+49)
      WRITE(6,2000) O,HEAD,(K,K=1,NDF)
      DO 100 I=N,J
      DO 75 K = 1,NDF
      R(K,I) = -R(K,I)
      RSUM(K) = RSUM(K) + R(K,I)
75    ASUM(K) = ASUM(K) + DABS(R(K,I))
100   WRITE(6,2001) I,(R(K,I),K=1,NDF)
C.... PRINT STATICS CHECK
      WRITE(6,2002) (RSUM(K),K=1,NDF)
      WRITE(6,2003) (ASUM(K),K=1,NDF)
      RETURN
2000  FORMAT(A1,20A4//5X,15HNODAL REACTIONS//6X,4HNODE,
```

```
     1  6(I9,4H DOF))
2001    FORMAT(I10,6E13.4)
2002    FORMAT(/7X,3HSUM,6E13.4)
2003    FORMAT(/3X,7HABS SUM,6E13.4)
        END


        SUBROUTINE PSETM(NA,NE,NJ,AFL)
        IMPLICIT REAL*8 (A-H,O-Z)
C
C.....  SET POINTER FOR ARRAYS
C
        LOGICAL AFL
        NA = NE
C**     NA = NA + MOD((NA-1),IPR)
        NE = NA + NJ
        AFL = .FALSE.
        CALL SETMEM(NE)
        RETURN
        END


        SUBROUTINE PZERO(V,NN)
C
C.....  ZERO REAL ARRAY
C
        DIMENSION V(NN)
        DO 100 N = 1,NN
100     V(N) = 0.0D0
        RETURN
        END


        SUBROUTINE UACTCL(A,C,B,JDIAG,NEQ,AFAC,BACK)
        IMPLICIT REAL*8 (A-H,O-Z)
        LOGICAL AFAC,BACK
        DIMENSION A(1),B(1),JDIAG(1),C(1)
C
C.....  UNSYMMETRIC, ACTIVE COLUMN PROFILE EQUATION SOLVER
C
C.....  FACTOR A TO UT*D*U, REDUCE B TO Y
        JR = 0
        DO 300 J = 1,NEQ
        JD = JDIAG(J)
        JH = JD - JR
        IF(JH.LE.1) GO TO 300
        IS = J + 1 - JH
        IE = J - 1
```

```
      IF(.NOT.AFAC) GO TO 250
      K = JR + 1
      ID = 0
C.... REDUCE ALL EQUATIONS EXCEPT DIAGONAL
      DO 200 I = IS,IE
      IR =ID
      ID = JDIAG(I)
      IH = MINO(ID - IR - 1,I - IS)
      IF(IH.EQ.0) GO TO 150
      A(K) = A(K) - DOT(A(K-IH),C(ID-IH),IH)
      C(K) = C(K) - DOT(C(K-IH),A(ID-IH),IH)
150   IF(A(ID).NE.0.0D0) C(K) = C(K)/A(ID)
200   K = K + 1
C.... REDUCE DIAGONAL TERM
      A(JD) = A(JD) - DOT(A(JR+1),C(JR+1),JH-1)
C.... FORWARD REDUCE THE R.H.S.
250   IF(BACK) B(J) = B(J) - DOT(C(JR+1),B(IS),JH-1)
300   JR=JD
      IF(.NOT.BACK) RETURN
C.... BACKSUBSTITUTION
      J = NEQ
      JD = JDIAG(J)
500   IF(A(JD).NE.0.0D0) B(J) = B(J)/A(JD)
      D = B(J)
      J = J - 1
      IF(J.LE.0) RETURN
      JR = JDIAG (J)
      IF(JD - JR.LE.1) GO TO 700
      IS = J - JD + JR + 2
      K= JR - IS + 1
      DO 600 I = IS,J
600   B(I) = B(I) - A(I+K)*D
700   JD = JR
      GO TO 500
      END
```

```
      SUBROUTINE PCDE1 (CT,XA,XC,XM,XN,U,F,JDIAG,ID,M,NE,NDF,AFR,CFR,DFR
     1  ,EFR,EER)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... FIRST ORDER ORDINARY DIFFERENTIAL EQUATION SOLVER
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      LOGICAL PCOMP,EER,AFR,CFR,DFR,EFR
      DIMENSION F(1),U(1),JDIAG(1),M(1),XA(1),XC(1),XM(1),XN(1),ID(1)
     1  ,CT(3),WD(3)
      DATA WD /4HINIT,4HLINE,4HQUAD/
      NNEQ = NDF*NUMNP
      REWIND 9
      WRITE (9) (M(I),I=1,NE)
      END FILE 9
      DO 100 J=1,3
      IF(PCOMP(CT(1),WD(J))) GO TO 110
100   CONTINUE
      EER = .TRUE.
      RETURN
110   EER = .FALSE.
      GO TO (1,2,3),J
1     DO 10 I=1,NNEQ
10    M(I) = ID(I)
      M1 = NNEQ + 1
      M1 = M1 + MOD((M1-1),IPR)
      CALL INIT (XA,XC,XM,XN,F,M,M(M1),U,JDIAG,NDF,NNEQ)
      GO TO 30
2     CALL PLINE (XA,XC,XM,XN,F,M,U,JDIAG,AFR,CFR,DFR,EFR)
      CALL TERM (XA,XC,XM,XN,F,M,U,JDIAG,AFR,CFR,DFR,EFR)
      GO TO 30
3     M2 = 1 + NEQ*IPR
      M2 = M2 + MOD((M2-1),IPR)
      CALL PQUAD (XA,XC,XM,XN,F,M,M(M2),U,JDIAG,AFR,CFR,DFR,EFR)
      CALL TERM (XA,XC,XM,XN,F,M(M2),U,JDIAG,AFR,CFR,DFR,EFR)
30    REWIND 9
      READ (9) (M(I),I=1,NE)
      RETURN
      END


      SUBROUTINE INIT (XA,XC,XM,XN,F,ID,UP1,U,JDIAG,NDF,NNEQ)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... INPUT INITIAL CONDITIONS
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /TDATA/ TIME,DT,C1,C2,C3,C4,C5
```

```
      COMMON /LDATA/ DUMAX,DUMIN,CONTR
      DIMENSION UI(3),XA(1),XC(1),XM(1),XN(1),F(1),UP1(1),U(1),ID(1),
     1  JDIAG(1)
      LOGICAL ERR,INIC,INPUT,AFR,CFR,DFR,EFR,CONTR
      DATA UI/4H INI,4HT CI,4HSPL /
      NN = JDIAG(NEQ)
      READ (5,1000) C5,C1,C2,DUMAX,DUMIN
      IF(C5.LE.0.0D0) C5 = 2.0D0/3.0D0
      IF(C1.LE.0.0D0) C1 = 1.5D0
      IF(C2.LE.0.0D0) C2 = .8D0
      WRITE (6,1002) C5,C1,C2
      CONTR =.TRUE.
      IF(DUMAX.LE.0.0D0.OR.DUMIN.LE.0.0D0) CONTR = .FALSE.
      IF(CONTR) WRITE (6,1001) DUMAX,DUMIN
      DO 9 I=1,NNEQ
9     UP1(I) = 0.0D0
      CALL GENVEC (NDF,UP1,UI,.TRUE.,ERR,.FALSE.)
      INPUT = .TRUE.
      DO 10 I=1,NNEQ
      K = IABS(ID(I))
      IF(K.GT.0) U(K) = UP1(I)
10    CONTINUE
      REWIND 10
      WRITE (10) (U(J),J=1,NEQ)
      END FILE 10
      RETURN
1000  FORMAT(5F10.0)
1001  FORMAT(5X,8HDUMAX = ,1PG13.5,8HDUMIN = ,1PG13.5)
1002  FORMAT(5X,8HTHETA = ,1PG13.5,8HGAMMA = ,1PG13.5,7HBETA = ,1PG13.5)
      END


      SUBROUTINE PLINE (XA,XC,XM,XN,F,UP1,U,JDIAG,AFR,CFR,DFR,EFR)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... TWO POINT SCHEME
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /TDATA/ TIME,DT,C1,C2,C3,C4,C5
      DIMENSION XA(1),XC(1),XM(1),XN(1),F(1),UP1(1),U(1),JDIAG(1)
      LOGICAL AFR,CFR,DFR,EFR
      NN = JDIAG(NEQ)
      REWIND 10
      WRITE (10) (XA(J),J=1,NN)
      END FILE 10
      DO 11 I=1,NEQ
11    UP1(I) = DT*F(I)
      CC = 1.0D0/(C5*DT)
```

```
      IF(DFR) CALL PROMUL (XM,U,UP1,JDIAG,NEQ)
      IF(EFR) CALL PRODIA (XN,U,UP1,NEQ)
      CALL PSCAL (UP1,CC,NEQ)
      CALL PROMUL (XA,U,UP1,JDIAG,NEQ)
      IF(DFR) CALL PADDM (XA,XM,CC,NN)
      IF(EFR) CALL PADDD (XA,XN,JDIAG,CC,NEQ)
      CALL ACTCOL (XA,UP1,JDIAG,NEQ,.TRUE.,.TRUE.)
      REWIND 10
      READ (10) (XA(J),J=1,NN)
      RETURN
      END


      SUBROUTINE PQUAD (XA,XC,XM,XN,F,UM1,UP1,U,JDIAG,AFR,CFR,DFR,EFR)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... THREE POINT SCHEME
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /TDATA/ TIME,DT,C1,C2,C3,C4,C5
      DIMENSION XA(1),XC(1),XM(1),XN(1),F(1),UM1(1),U(1),UP1(1),JDIAG(1)
      LOGICAL AFR,CFR,DFR,EFR,CONTR
      REWIND 10
      READ (10)  (UM1(J),J=1,NEQ)
      NN = JDIAG(NEQ)
      REWIND 10
      WRITE (10) (XA(J),J=1,NN),(F(J),J=1,NEQ)
      END FILE 10
      C3 = C2*DT
      C4 = C1/C3
      DO 10 I=1,NEQ
      UP1(I) = 1.0D0/C2*F(I)
      F(I) = ((2.0D0*C1-1.0D0)/C3)*U(I)
      F(I) = F(I) + ((1.0D0-C1)/C3)*UM1(I)
10    CONTINUE
      IF(DFR) CALL PROMUL (XM,F,UP1,JDIAG,NEQ)
      IF(EFR) CALL PRODIA (XN,F,UP1,NEQ)
      DO 11 I=1,NEQ
      F(I) = (2.0D0+(.5D0-C1)/C2)*U(I)
      F(I) = F(I) + ((C1-.5D0)/C2-1.0D0)*UM1(I)
11    CONTINUE
      CALL PROMUL (XA,F,UP1,JDIAG,NEQ)
      IF(DFR) CALL PADDM (XA,XM,C4,NN)
      IF(EFR) CALL PADDD (XA,XN,JDIAG,C4,NEQ)
      CALL ACTCOL (XA,UP1,JDIAG,NEQ,.TRUE.,.TRUE.)
      REWIND 10
      READ (10) (XA(J),J=1,NN),(F(J),J=1,NEQ)
      RETURN
```

```
      END

      SUBROUTINE TERM (XA,XC,XM,XN,F,UP1,U,JDIAG,AFR,CFR,DFR,EFR)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /TDATA/ TIME,DT,C1,C2,C3,C4,C5
      DIMENSION XA(1),XC(1),XM(1),XN(1),F(1),U(1),UP1(1),JDIAG(1)
      COMMON /UDATA/ DUMAX,DUMIN,CONTR
      LOGICAL AFR,CFR,DFR,EFR,CONTR
      IF(.NOT.CONTR) GO TO 3
10    UN = 0.0D0
      DO 11 JJ = 1,NEQ
11    UN = UN + (UP1(JJ)-U(JJ))*(UP1(JJ)-U(JJ))
      UN = DSQRT (UN)
      K = 3
      IF(UN.GT.DUMAX) K = 1
      IF(UN.LT.DUMIN) K = 2
      GO TO (1,2,3),K
1     DT = DT/2.0D0
      CALL PLINE (XA,XC,XM,XN,F,UP1,U,JDIAG,AFR,CFR,DFR,EFR)
      GO TO 10
2     DT = DT*2.0D0
      CALL PLINE (XA,XC,XM,XN,F,UP1,U,JDIAG,AFR,CFR,DFR,EFR)
3     REWIND 10
      WRITE (10) (U(J),J=1,NEQ)
      END FILE 10
      DO 14 I=1,NEQ
14    U(I) = UP1(I)
      WRITE(6,1000) DT
1000  FORMAT(10X,4HDT =,1PG13.6)
      RETURN
      END
```

```
      SUBROUTINE ELMT02(D,UL,XL,IX,TL,S,P,NDF,NDM,NST,ISW)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... TWO DIMENSIONAL HEAT TRANSFER ELEMENT
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON/ELDATA/ DM,N,MA,MCT,IEL,NEL
      DIMENSION D(1),UL(1),XL(NDM,1),IX(1),TL(1),S(NST,1),P(1),
     1SHP(3,9),WLAB(4)
      DATA WLAB/4H PLA,4HNE  ,4HAXIS,4HYM  /
C
C.... TRANSFER TO CORRECT PROCESSOR
      GO TO (1,2,3,2,5,3),ISW
C
C.... INPUT MATERIALS PROPERTIES
1     NLBC = 0
      READ(5,1000) D(1),D(2),D(3),D(4),D(5),NGP,KAT,NLBC,INOL
      IF(NGP.LE.0.OR.NGP.GT.6) NGP = 4
      IF(KAT.NE.2) KAT = 1
      WRITE(6,2000) D(1),D(2),D(3),D(4),D(5),NGP,NLBC
      D(3) = D(3)*D(4)
      IF(KAT.EQ.2) WRITE(6,2001) WLAB(3),WLAB(4)
      IF(KAT.EQ.1) WRITE(6,2001) WLAB(1),WLAB(2)
      IKX = 0
      IKY = 0
      IRCC = 0
      IQ = 0
      IF(INOL.EQ.0) GO TO 11
      READ (5,1001) IKX,IKY,IRCC,IQ
      IF(IKX.NE.0) CALL PKX (T,D(1),1)
      IF(IKY.NE.0) CALL PKY (T,D(2),1)
      IF(IRCC.NE.0) CALL PRCC (T,D(3),1)
      IF(IQ.NE.0) CALL PQ (T,D(5),1)
11    CONTINUE
      IF(NLBC.NE.0) CALL BCOND2 (UL,XL,IX,S,P,NDF,NDM,NST,NLBC,NGP,1)
      RETURN
C
C.... INSERT CHECK OF MESH IF DESIRED
2     RETURN
C
C.... COMPUTE CONDUCTIVITY (STIFFNESS) MATRIX
3     DO 102 II=1,NGP
      CALL PGAUSS(SG,WS,NGP,II)
      DO 102 JJ=1,NGP
      CALL PGAUSS(TG,WT,NGP,JJ)
      WG = WS*WT
      CALL SHAPE(SG,TG,XL,SHP,XSJ,NDM,NEL,IX,.FALSE.)
      IF(IKX.EQ.0.AND.IKY.EQ.0.AND.IQ.EQ.0) GO TO 113
```

```
      T = 0.0D0
      DO 111 KK=1,NEL
111   T = T + SHP(3,KK)*UL(KK)
      IF(IKX.NE.0) CALL PKX(T,D(1),2)
      IF(IKY.NE.0) CALL PKY(T,D(2),2)
      IF(IQ.NE.0) CALL PQ(T,D(5),2)
113   CONTINUE
      IF(KAT.NE.2) GO TO 101
      RR = 0.0D0
      DO 100 I = 1,NEL
100   RR = RR + SHP(3,I)*XL(1,I)
      XSJ = XSJ*RR
101   DV = XSJ*WG
      DO 102 J = 1,NEL
      SHJ = SHP(3,J)*DV
      A1 = D(1)*SHP(1,J)*DV
      A2 = D(2)*SHP(2,J)*DV
      P(J) = P(J) + D(5)*SHJ
      DO 102 I = 1,NEL
102   S(I,J) = S(I,J) + A1*SHP(1,I) + A2*SHP(2,I)
      IF(NLBC.NE.0) CALL BCOND2 (UL,XL,IX,S,P,NDF,NDM,NST,NLBC,NGP,2)
      DO 106 I =1,NEL
      DO 106 J = 1,NEL
106   P(I) = P(I) - S(I,J) * UL(J)
      IF(ISW.EQ.3) RETURN
      DO 116 I=1,NEL
      DO 116 J=1,NEL
116   S(I,J) = 0.0D0
C
C.... COMPUTE HEAT CAPACITY (MASS) MATRIX
5     DO 105 II = 1,NGP
      CALL PGAUSS(SG,WS,NGF,II)
      DO 105 JJ=1,NGP
      CALL PGAUSS(TG,WT,NGF,JJ)
      WG = WS*WT
      CALL SHAPE(SG,TG,XL,SHP,XSJ,NDM,NEL,IX,.FALSE.)
      IF(IRCC.EQ.0) GO TO 114
      T = 0.0D0
      DO 112 KK=1,NEL
112   T = T + SHP(3,KK)*UL(KK)
      CALL PRCC(T,D(3),2)
114   CONTINUE
      IF(KAT.NE.2) GO TO 104
      RR = 0.0D0
      DO 103 I = 1,NEL
103   RR = RR + SHP(3,I)*XL(1,I)
      XSJ = XSJ*RR
104   DV = XSJ*WG
```

```
      DO 105 J = 1,NEL
      SHJ = D(3)*SHP(3,J)*DV
      IF(ISW.EQ.5) P(J) = P(J) + SHJ
      DO 105 I = 1,NEL
105   S(I,J) = S(I,J) + SHJ*SHP(3,I)
      IF(ISW.EQ.5) RETURN
      DO 601 I=1,NEL
      DO 601 J=1,NEL
601   P(I) = P(I) - S(I,J)*TL(J)
      RETURN
C
C.... FORMATS
1000  FORMAT(5F10.0,4I5)
1001  FORMAT (4I5)
2000  FORMAT(5X,26HHEAT CONDUCTION ELEMENT    // 7X,
     1  12HCONDUCTIVITY ,5X,4HKX = ,1PG15.7, 5X,4HKY =,1PG15.7,
     2  /7X,19HSPECIFIC HEAT         ,1PG15.7,/7X,
     3  19HDENSITY             ,1PG15.7,/7X,
     4  19HHEAT GENER/UNIT VOL,1PG15.7/,/7X,
     5  I3,15H  GAUSS PTS/DIR,/7X,
     6  I3,42H  LINES WITH SPECIFIED BOUNDARY CONDITIONS)
2001  FORMAT(6X,2A4,9H ANALYSIS )
      END


      SUBROUTINE SHAPE(SS,TT,X,SHP,XSJ,NDM,NEL,IX,FLG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... SHAPE FUNCTION ROUTINE FOR TWO DIMENSIONAL ELEMENTS
C
      LOGICAL FLG
      COMMON /JAC2L/ XS
      DIMENSION SHP(3,1),X(NDM,1),S(4),T(4),XS(2,2),SX(2,2),IX(1)
      DATA S/-0.5D0,0.5D0,0.5D0,-0.5D0/,T/-0.5D0,-0.5D0,0.5D0,0.5D0/
C.... FORM 4-NODE QUADRILATERAL SHAPE FUNCTIONS
      DO 100 I = 1,4
      SHP(3,I) = (0.5D0+S(I)*SS)*(0.5D0+T(I)*TT)
      SHP(1,I) = S(I)*(0.5D0+T(I)*TT)
100   SHP(2,I) = T(I)*(0.5D0+S(I)*SS)
      IF(NEL.GE.4) GO TO 120
C.... FORM TRIANGLE BY ADDING THIRD AND FOURTH TOGETHER
      DO 110 I = 1,3
110   SHP(I,3) = SHP(I,3)+SHP(I,4)
C.... ADD QUADRATIC TERMS IF NECESSARY
120   IF(NEL.GT.4) CALL SHAP2(SS,TT,SHP,IX,NEL)
C.... CONSTRUCT JACOBIAN AND ITS INVERSE
      DO 130 I = 1,NDM
      DO 130 J = 1,2
```

```
      XS(I,J) = 0.0D0
      DO 130 K = 1,NEL
130   XS(I,J) = XS(I,J) + X(J,K)*SHP(I,K)
      XSJ = XS(1,1)*XS(2,2)-XS(1,2)*XS(2,1)
      IF(FLG) RETURN
      SX(1,1) = XS(2,2)/XSJ
      SX(2,2) = XS(1,1)/XSJ
      SX(1,2) =-XS(1,2)/XSJ
      SX(2,1) =-XS(2,1)/XSJ
C.... FORM GLOBAL DERIVATIVES
      DO 140 I = 1,NEL
      TP       = SHP(1,I)*SX(1,1)+SHP(2,I)*SX(1,2)
      SHP(2,I) = SHP(1,I)*SX(2,1)+SHP(2,I)*SX(2,2)
140   SHP(1,I) = TP
      RETURN
      END


      SUBROUTINE SHAP2(S,T,SHP,IX,NEL)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... ADD QUADRATIC FUNCTIONS AS NECESSARY
C
      DIMENSION IX(1),SHP(3,1)
      S2 = (1.0D0-S*S)/2.0D0
      T2 = (1.0D0-T*T)/2.0D0
      DO 100 I = 5,9
      DO 100 J = 1,3
100   SHP(J,I) = 0.0D0
C.... MIDSIDE NODES (SERENDIPITY)
      IF(IX(5).EQ.0) GO TO 101
      SHP(1,5) = -S*(1.0D0-T)
      SHP(2,5) = -S2
      SHP(3,5) = S2*(1.0D0-T)
101   IF(NEL.LT.6) GO TO 107
      IF(IX(6).EQ.0) GO TO 102
      SHP(1,6) = T2
      SHP(2,6) = -T*(1.0D0+S)
      SHP(3,6) = T2*(1.0D0+S)
102   IF(NEL.LT.7) GO TO 107
      IF(IX(7).EQ.0) GO TO 103
      SHP(1,7) = -S*(1.0D0+T)
      SHP(2,7) = S2
      SHP(3,7) = S2*(1.0D0+T)
103   IF(NEL.LT.8) GO TO 107
      IF(IX(8).EQ.0) GO TO 104
      SHP(1,8) = -T2
      SHP(2,8) = -T*(1.0D0-S)
```

```
      SHP(3,8) = T2*(1.0D0-S)
C.... INTERIOR NODE (LAGRANGIAN)
104   IF(NEL.LT.9) GO TO 107
      IF(IX(9).EQ.0) GO TO 107
      SHP(1,9) = -S*T2
      SHP(2,9) = -T*S2
      SHP(3,9) = 4.0D0*S2*T2
C.... CORRECT EDGE NODES FOR INTERIOR NODE (LAGRANGIAN)
      DO 106 J = 1,3
      DO 105 I = 1,4
105   SHP(J,I) = SHP(J,I) - 0.25D0*SHP(J,9)
      DO 106 I = 5,8
106   IF(IX(I).NE.0) SHP(J,I) = SHP(J,I) - 0.5D0*SHP(J,9)
C.... CORRECT CORNER NODES FOR PRESENCE OF MIDSIDE NODES
107   K = 8
      DO 109 I = 1,4
      L = I + 4
      DO 108 J = 1,3
108   SHP(J,I) = SHP(J,I) -0.5D0*(SHP(J,K)+SHP(J,L))
109   K = L
      RETURN
      END


      SUBROUTINE BCOND2 (UL,XL,IX,S,P,NDF,NDM,NST,NLBC,NGP,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /ELDATA/ DM,N,MA,MCT,IEL,NEL
      DIMENSION UL(1),XL(NDM,1),IX(1),S(NST,1),P(1),SHP(3,9)
      DIMENSION KEL(50),KBCOND(50),KLINE(50),PROPB(50,2)
      GO TO (1,2),IFLAG
C
C.... INPUT LINE BOUNDARY CONDITIONS
1     WRITE(6,2000) O,HEAD
      KBMAX = 0
      DO 10 I=1,NLBC
      READ (5,1000) KEL(I),KBCOND(I),KLINE(I),PROPB(I,1),PROPB(I,2)
      KBMAX = MAX0(KBMAX,KBCOND(I))
10    WRITE(6,2001) KEL(I),KBCOND(I),KLINE(I),PROPB(I,1),PROPB(I,2)
      IF(KBMAX.GE.4) CALL CONV(T,COEFF,1)
      RETURN
C
C.... ADD LINE BOUNDARY CONDITIONS CONTIBUITIONS
C.... TO CONDUCTIVITY MATRIX AND LOAD VECTOR
2     DO 30 LL=1,NLBC
      IF(KEL(LL).NE.N) GO TO 30
      KBT = KBCOND(LL)
      KBL = KLINE(LL)
```

```
      CONST = PROPB(LL,1)
      TEMP =PROPE(LL,2)
      KBLABS = IABS(KBL)
      C2 = DFLOAT(KBL/KBLABS)
      DO 30 II=1,NGP
      CALL PGAUSS (C1,W1,NGP,II)
      WG = W1
C
C.... SELECT CORRECT INTEGRATION POINTS
      GO TO (21,22),KBLABS
21    SG = C2
      TG = C1
      GO TO 24
22    SG = C1
      TG = C2
      GO TO 24
24    CALL SHAPE (SG,TG,XL,SHP,DUM,NDM,NEL,IX,.TRUE.)
      CALL JACOB2 (XSJ,KBLABS)
      DS = WG*XSJ
      COEFF = CONST
      GO TO (25,26,27,28),KBT
C
C.... FLUX BOUNDARY CONDITION
25    DO 251 L=1,NEL
251   P(L) = P(L) - COEFF*SHP(3,L)*DS
      GO TO 30
C
C.... RADIATION BOUNDARY CONDITION
27    T = 0.0D0
      DO 271 I=1,NEL
271   T = T + SHP(3,I)*UL(I)
      COEFF = COEFF*(T*T + TEMP*TEMP)*(T + TEMP)
      GO TO 26
C
C.... CONVECTION BOUNDARY CONDITION (TEMP. DEPEDENT)
28    T = 0.0D0
      DO 281 I=1,NEL
281   T = T + SHP(3,I)*UL(I)
      CALL CONV(T,COEFF,2)
C
C.... CONVECTION BOUNDARY CONDITION
26    DO 261 J=1,NEL
      A = SHP(3,J)*DS
      P(J) = P(J) + COEFF*TEMP*A
      DO 261 I = 1,NEL
261   S(I,J) = S(I,J) + COEFF*SHP(3,I)*A
30    CONTINUE
      RETURN
```

```
1000  FORMAT (3I5,2F10.0)
2000  FORMAT (A1,20A4,//5X,9HLINE B.C.//,
     15X,52H ELEM     B.C.     LINE   PROPERTY VALUE        TEMPERATURE)
2001  FORMAT (5X,3(I5,2X),2(1PG15.7,2X))
      END


      SUBROUTINE JACBB2 (XSJ,KL)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... COMPUTE LINE JACOBIAN DETERMINANT
C
      COMMON /JAC2L/ XS
      DIMENSION XS(2,2)
      GO TO (1,2),KL
1     D1 = XS(2,2)
      D2 = -XS(2,1)
      GO TO 10
2     D1 = XS(1,1)
      D2 = -XS(1,2)
      GO TO 10
10    XSJ = DSQRT(D1*D1 + D2*D2 )
      RETURN
      END


      SUBROUTINE PKX (T,C,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CALCULATE CONDUCTIVITY (X DIRECTION)
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION TEMP(50),COEF(50)
      IF(IFLAG.EQ.1) WRITE (6,2000) O,HEAD
      CALL TABLE (TEMP,COEF,T,C,NDATA,IFLAG)
      RETURN
2000  FORMAT(A1,20A4,//5X,26HCONDUCTIVITY DATA (X DIR.)//,
     1  5X,30H   TEMPERATURE                KX     )
      END


      SUBROUTINE PKY (T,C,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CALCULATE CONDUCTIVITY (Y DIRECTION)
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION TEMP(50),COEF(50)
```

```
      IF(IFLAG.EQ.1) WRITE (6,2000) O,HEAD
      CALL TABLE (TEMP,COEF,T,C,NDATA,IFLAG)
      RETURN
 2000 FORMAT(A1,20A4,//5X,26HCONDUCTIVITY DATA (Y DIR.)//,
     1  5X,30H   TEMPERATURE              KY      )
      END


      SUBROUTINE PROC (T,C,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CALCULATE HEAT CAPACITY
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION TEMP(50),COEF(50)
      IF(IFLAG.EQ.1) WRITE (6,2000) O,HEAD
      CALL TABLE (TEMP,COEF,T,C,NDATA,IFLAG)
      RETURN
 2000 FORMAT(A1,20A4,//5X,18HHEAT CAPACITY DATA//,
     1  5X,30H   TEMPERATURE          RO * C   )
      END


      SUBROUTINE PC (T,C,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CALCULATE INTERNAL HEAT GENERATION
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION TEMP(50),COEF(50)
      IF(IFLAG.EQ.1) WRITE (6,2000) O,HEAD
      CALL TABLE (TEMP,COEF,T,C,NDATA,IFLAG)
      RETURN
 2000 FORMAT(A1,20A4,//5X,20HINT.HEAT GENER. DATA//,
     1  5X,30H   TEMPERATURE              Q       )
      END


      SUBROUTINE CONV (T,C,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CALCULATE HEAT TRANSFER COEFFICIENT
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION TEMP(50),COEF(50)
      IF(IFLAG.EQ.1) WRITE (6,2000) O,HEAD
      CALL TABLE (TEMP,COEF,T,C,NDATA,IFLAG)
      RETURN
```

```
2000  FORMAT(A1,20A4,//5X,24HHEAT TRANSFER COEF. CATA//,
     1  5X,30H  TEMPERATURE      COEFFICIENT)
      END


      SUBROUTINE TABLE (XX,YY,X,Y,N,K)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CALCULATE Y =TABLE(X) BY LINEAR INTERPOLATION
C
      DIMENSION XX(1),YY(1)
      GO TO(1,2),K
1     READ (5,1000) N
      DO 11 I=1,N
      READ (5,1001) XX(I),YY(I)
11    WRITE(6,2001) XX(I),YY(I)
      RETURN
2     IF(X.LT.XX(N).AND.X.GT.XX(1)) GO TO 21
      IF(X.GE.XX(N)) Y = YY(N)
      IF(X.LE.XX(1)) Y = YY(1)
      RETURN
21    DO 22 I=1,N
      IF(X.LE.XX(I)) GO TO 23
22    CONTINUE
23    X1 = XX(I) - X
      X2 = XX(I-1) - X
      Y1 = YY(I)
      Y2 = YY(I-1)
      Y = (X1*Y2 - X2*Y1)/(X1 - X2)
      RETURN
1000  FORMAT(I5)
1001  FORMAT(2F10.0)
2001  FORMAT(5X,2(1PG15.7,2X))
      END
```

```
      SUBROUTINE ELMT03 (D,UL,XL,IX,TL,S,P,NDF,NDM,NST,ISW)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... THREE DIMENSIONAL HEAT TRANFER ELEMENT
C
C           C           MATERIAL PROPERTIES
C              D(1) CONDUCTIVITY IN X DIRECTION
C              D(2) CONDUCTIVITY IN Y DIRECTION
C              D(3) CONDUCTIVITY IN Z DIRECTION
C              D(4) SPECIFIC HEAT
C              D(5) MASS DENSITY
C              D(6) HEAT GENERATION PER UNIT VOLUME
C           NGP         NUMBER OF INTEGRATION POINTS
C           WG          INTEGRATION WEIGHT
C           KAT         =2 FOR AXISYMMETRY
C                       =1 FOR PLANE GEOMETRY
C           NSBC        NUMBER OF SURFACES WITH SPECIFIED
C                       BOUNDARY CONDITIONS
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /ELDATA/ DM,N,MA,MCT,IEL,NEL
      DIMENSION D(1),UL(1),XL(NDM,1),IX(1),TL(1),S(NST,1),P(1),
     1  SHP(4,21),WLAB(4)
      DATA WLAB /4H PLA,4HNE  ,4HAXIS,4HYM  /
C
C.... TRANFER TO CORRECT PROCESSOR
      GO TO (1,2,3,2,5,3), ISW
C
C.... INPUT MATERIAL PROPERTIES
1     NSBC = 0
      READ(5,1000) D(1),D(2),D(3),D(4),D(5),D(6),NGP,KAT,NSBC,INOL
      IF(NGP.LE.0.OR.NGP.GT.6) NGP = 4
      WRITE(6,2000) D(1),D(2),D(3),D(4),D(5),D(6),NGP,NSBC
      D(4) = D(4)*D(5)
      IF(KAT.NE.2) KAT = 1
      IF (KAT.EQ.1) WRITE(6,2001) WLAB(1),WLAB(2)
      IF (KAT.EQ.2) WRITE(6,2001) WLAB(3),WLAB(4)
      IF(INOL.EQ.0) GO TO 11
      READ(5,1001) IKX,IKY,IKZ,IROC,IQ
      IF(IKX.NE.0) CALL PKX (T,D(1),1)
      IF(IKY.NE.0) CALL PKY (T,D(2),1)
      IF(IKZ.NE.0) CALL PKZ(T,D(3),1)
      IF(IROC.NE.0) CALL PROC (T,D(4),1)
      IF(IQ.NE.0) CALL PQ  (T,D(6),1)
11    CONTINUE
      IF(NSBC.NE.0) CALL BCOND3 (UL,XL,IX,S,P,NDF,NDM,NST,NSBC,NGP,1)
      RETURN
C
```

```
C.... INSERT CHECK OF MESH IF DESIRED
2     RETURN
C
C.... COMPUTE CONDUCTIVITY (STIFFNESS) MATRIX
3     DO 102 II=1,NGP
      CALL PGAUSS(RG,WR,NGP,II)
      DO  102 JJ=1,NGP
      CALL PGAUSS(SG,WS,NGP,JJ)
      DO 102 KK=1,NGP
      CALL PGAUSS(TG,WT,NGP,KK)
      WG = WR*WS*WT
      CALL SHAP3D(RG,SG,TG,XL,SHP,XSJ,NDM,NEL,IX,.FALSE.)
      IF(IKX.EQ.0.AND.IKY.EQ.0.AND.IKZ.EQ.0.AND.IQ.EQ.0) GO TO 112
      T = 0.0D0
      DO 111 L=1,NEL
111   T = T + SHP(4,L)*UL(L)
      IF(IKX.NE.0) CALL PKX(T,D(1),2)
      IF(IKY.NE.0) CALL PKY(T,D(2),2)
      IF(IKZ.NE.0) CALL PKZ(T,D(3),2)
      IF(IQ.NE.0)   CALL PQ(T,D(6),2)
112   CONTINUE
      IF(KAT.NE.2) GO TO 101
      RR = 0.0D0
      DO 100 I=1,NEL
100   RR = RR + SHP(4,I)*XL(1,I)
      XSJ = XSJ*RR
101   DV = XSJ*WG
      D11 = D(1)*DV
      D12 = D(2)*DV
      D13 = D(3)*DV
      DO 102 J=1,NEL
      DB11 = D11*SHP(1,J)
      DB21 = D12*SHP(2,J)
      DB31 = D13*SHP(3,J)
      P(J) = P(J) + SHP(4,J)*D(6)*DV
      DO 102 I = 1,NEL
102   S(I,J) = S(I,J) + SHP(1,I)*DB11 + SHP(2,I)*DB21 + SHP(3,I)*DB31
      IF(NSBC.NE.0) CALL BCOND3 (UL,XL,IX,S,P,NDF,NDM,NST,NSBC,NGP,2)
      DO 103 I=1,NEL
      DO 103 J=1,NEL
103   P(I) = P(I) - S(I,J)*UL(J)
      RETURN
C
C.... COMPUTE HEAT CAPACITY (MASS) MATRIX
5     DO 502 II=1,NGP
      CALL PGAUSS(RG,WR,NGP,II)
      DO 502 JJ=1,NGP
      CALL PGAUSS(SG,WS,NGP,JJ)
```

```
      DO 502 KK=1,NGP
      CALL PGAUSS(TG,WT,NGF,KK)
      WG = WR*WS*WT
      CALL SHAP3D(RG,SG,TG,XL,SHP,XSJ,NDM,NEL,IX,.FALSE.)
      IF(IPCC.EQ.0) GO TO 512
      T = 0.000
      DO 511 L=1,NEL
511   T = T + SHP(4,L)*UL(L)
      CALL PROC (T,D(4),2)
512   CONTINUE
      IF(KAT.NE.2) GO TO 501
      RR = 0.0D0
      DO 500 I=1,NEL
500   RR = RR + SHP(4,I)*XL(1,I)
      XSJ = XSJ*RR
501   DV = XSJ*WG
      DO 502 J=1,NEL
      SHJ = D(4)*SHP(4,J)*DV
      P(J) = P(J) + SHJ
      DO 502 I=1,NEL
502   S(I,J) = S(I,J) + SHJ*SHP(4,I)
      RETURN
1000  FORMAT(6F10.0,4I5)
1001  FORMAT(5I5)
2000  FORMAT(5X,23HHEAT CONDUCTION ELEMENT    // 7X,
     1  12HCONDUCTIVITY ,5X,4HKX =,1PG15.7,5X,4HKY =,1PG15.7,5X,
     2  4HKZ =,1PG15.7,/7X,19HSPECIFIC HEAT          ,1PG15.7,/7X,
     3  19HDENSITY              ,1PG15.7,/7X,
     4  19HHEAT GENER/UNIT VOL,1PG15.7/,/7X,I3,15H  GAUSS PTS/DIR,/
     5  7X,I3,45H  SURFACES WITH SPECIFIED BOUNDARY CONDITIONS)
2001  FORMAT(6X,2A4,9H ANALYSIS )
      END

      SUBROUTINE SHAP3D (RR,SS,TT,X,SHP,XSJ,NDM,NEL,IX,FLG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... SHAPE FUNCTION ROUTINE FOR THREE DIMENSIONAL ELEMENTS
C.... 4-21 NODES
C
C            RR,SS,TT    NATURAL COORDINATES WHERE INTERPOLATION
C                        FUNCTIONS ARE EVALUATED
C            SHP(1,I)    X DERIVATIVE OF SHAPE FUNCTIONS
C            SHP(2,I)    Y DERIVATIVE OF SHAPE FUNCTIONS
C            SHP(3,I)    Z DERIVATIVE OF SHAPE FUNCTIONS
C            SHP(4,I)    SHAPE FUNCTIONS
C            XS          JACOBIAN ARRAY
C            XSJ         JACOBIAN DETERMINANT
```

```
C               X(1,I)      X NODAL COORDINATES
C               X(2,I)      Y NODAL COORDINATES
C               X(3,I)      Z NODAL COORDINATES
C
      LOGICAL FLG
      COMMON /JAC3S/ XS
      DIMENSION SHP(4,1),X(NDM,1),R(8),S(8),T(8),XS(3,3),SX(3,3),IX(1)
      DATA R/0.5D0,-0.5D0,-0.5D0,0.5D0,0.5D0,-0.5D0,-0.5D0,0.5D0/,
     1     S/0.5D0,0.5D0,-0.5D0,-0.5D0,0.5D0,0.5D0,-0.5D0,-0.5D0/,
     2     T/0.5D0,0.5D0,0.5D0,0.5D0,-0.5D0,-0.5D0,-0.5D0,-0.5D0/
C
C.... COMPUTE 8-NODE RECTANGULAR PRISM SHAPE FUNCTIONS AND DERIVATIVES
C.... IN NATURAL COORDINATES
      DO 100 I = 1,8
      SHP(4,I) = (0.5D0+R(I)*RR)*(0.5D0+S(I)*SS)*(0.5D0+T(I)*TT)
      SHP(1,I) = R(I)*(0.5D0+S(I)*SS)*(0.5D0+T(I)*TT)
      SHP(2,I) = S(I)*(0.5D0+R(I)*RR)*(0.5D0+T(I)*TT)
100   SHP(3,I) = T(I)*(0.5D0+R(I)*RR)*(0.5D0+S(I)*SS)
      IF(NEL.GE.8) GO TO 200
C
C.... DEGENERATE FORMS
C
C.... ADD QUADRATIC TERMS IF NECESSARY
200   IF(NEL.GT.8) CALL SHAP32(RR,SS,TT,SHP,IX,NEL)
C
C.... CONSTRUCT JACOBIAN AND ITS INVERSE
201   DO 300 I = 1,NDM
      DO 300 J = 1,3
      XS(I,J) = 0.0D0
      DO 300 K = 1,NEL
300   XS(I,J) = XS(I,J) + X(J,K)*SHP(I,K)
      IF(FLG) RETURN
      XSJ = XS(1,1)*XS(2,2)*XS(3,3)
     1     +XS(1,2)*XS(2,3)*XS(3,1)
     2     +XS(1,3)*XS(2,1)*XS(3,2)
     3     -XS(1,3)*XS(2,2)*XS(3,1)
     4     -XS(1,2)*XS(2,1)*XS(3,3)
     5     -XS(1,1)*XS(2,3)*XS(3,2)
      IF(XSJ.GT.1.0D-13) GO TO 400
      WRITE (6,1000)
      STOP
400   SX(1,1) = (XS(2,2)*XS(3,3)-XS(3,2)*XS(2,3))/XSJ
      SX(2,1) =-(XS(2,1)*XS(3,3)-XS(3,1)*XS(2,3))/XSJ
      SX(3,1) = (XS(2,1)*XS(3,2)-XS(3,1)*XS(2,2))/XSJ
      SX(1,2) =-(XS(1,2)*XS(3,3)-XS(3,2)*XS(1,3))/XSJ
      SX(2,2) = (XS(1,1)*XS(3,3)-XS(3,1)*XS(1,3))/XSJ
      SX(3,2) =-(XS(1,1)*XS(3,2)-XS(3,1)*XS(1,2))/XSJ
      SX(1,3) = (XS(1,2)*XS(2,3)-XS(2,2)*XS(1,3))/XSJ
```

```
      SX(2,3) =-(XS(1,1)*XS(2,3)-XS(2,1)*XS(1,3))/XSJ
      SX(3,3) = (XS(1,1)*XS(2,2)-XS(2,1)*XS(1,2))/XSJ
C
C.... FORM GLOBAL DERIVATIVES
      DO 500 I =1,NEL
      TP1 = SHP(1,I)*SX(1,1) + SHP(2,I)*SX(1,2) + SHP(3,I)*SX(1,3)
      TP2 = SHP(1,I)*SX(2,1) + SHP(2,I)*SX(2,2) + SHP(3,I)*SX(2,3)
      SHP(3,I) = SHP(1,I)*SX(3,1) + SHP(2,I)*SX(3,2) + SHP(3,I)*SX(3,3)
      SHP(1,I) = TP1
500   SHP(2,I) = TP2
      RETURN
1000  FORMAT(5X,'**FATAL ERROR   ** NEGATIVE OR ZERO JACOBIAN DETERMINAN
     1T')
      END


      SUBROUTINE SHAP32(R,S,T,SHP,IX,NEL)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... ADD QUADRATIC FUNCTIONS AS NECESSARY
C
      DIMENSION IX(1),SHP(4,1)
      DATA ONE/1.0D0/,PFIVE/0.5D0/
      R2 = (ONE-R*R)/4.0D0
      S2 = (ONE-S*S)/4.0D0
      T2 = (ONE-T*T)/4.0D0
      DO 100 I = 9,21
      DO 100 J = 1,4
100   SHP(J,I) = 0.0D0
C
C.... MIDSIDE NODES (SERENDIPITY)
      IF(IX(9).EQ.0) GO TO 101
      SHP(1,9) = -R*PFIVE*(ONE+S)*(ONE+T)
      SHP(2,9) = R2*(ONE+T)
      SHP(3,9) = R2*(ONE+S)
      SHP(4,9) = R2*(ONE+S)*(ONE+T)
101   IF(NEL.LT.10) GO TO 200
      IF(IX(10).EQ.0) GO TO 102
      SHP(1,10) = -S2*(ONE+T)
      SHP(2,10) = -S*PFIVE*(ONE-R)*(ONE+T)
      SHP(3,10) = S2*(ONE-R)
      SHP(4,10) = S2*(ONE-R)*(ONE+T)
102   IF(NEL.LT.11) GO TO 200
      IF(IX(11).EQ.0) GO TO 103
      SHP(1,11) = -R*PFIVE*(ONE-S)*(ONE+T)
      SHP(2,11) = -R2*(ONE+T)
      SHP(3,11) = R2*(ONE-S)
```

```
      SHP(4,11) = R2*(ONE-S)*(ONE+T)                               ...02241
103   IF(NEL.LT.12) GO TO 200                                      ...02242
      IF(IX(12).EQ.0) GO TO 104                                    ...02243
      SHP(1,12) = S2*(ONE+T)                                       ...02244
      SHP(2,12) = -S*PFIVE*(ONE+R)*(ONE+T)                         ...02245
      SHP(3,12) = S2*(ONE+R)                                       ...02246
      SHP(4,12) = S2*(ONE+R)*(ONE+T)                               ...02247
104   IF(NEL.LT.13) GO TO 200                                      ...02248
      IF(IX(13).EQ.0) GO TO 105                                    ...02249
      SHP(1,13) = -R*PFIVE*(ONE+S)*(ONE-T)                         ...02250
      SHP(2,13) = R2*  (ONE-T)                                     ...02251
      SHP(3,13) = -R2*(ONE+S)                                      ...02252
      SHP(4,13) = R2*(ONE+S)*(ONE-T)                               ...02253
105   IF(NEL.LT.14) GO TO 200                                      ...02254
      IF(IX(14).EQ.0) GO TO 106                                    ...02255
      SHP(1,14) = -S2*(ONE-T)                                      ...02256
      SHP(2,14) = -S*PFIVE*(ONE-R)*(ONE-T)                         ...02257
      SHP(3,14) = -S2*(ONE-R)                                      ...02258
      SHP(4,14) = S2*(ONE-R)*(ONE-T)                               ...02259
106   IF(NEL.LT.15) GO TO 200                                      ...02260
      IF(IX(15).EQ.0) GO TO 107                                    ...02261
      SHP(1,15) = -R*PFIVE*(ONE-S)*(ONE-T)                         ...02262
      SHP(2,15) = -R2*(ONE-T)                                      ...02263
      SHP(3,15) = -R2*(ONE-S)                                      ...02264
      SHP(4,15) = R2*(ONE-S)*(ONE-T)                               ...02265
107   IF(NEL.LT.16) GO TO 200                                      ...02266
      IF(IX(16).EQ.0) GO TO 108                                    ...02267
      SHP(1,16) = S2*(ONE-T)                                       ...02268
      SHP(2,16) = -S*PFIVE*(ONE+R)*(ONE-T)                         ...02269
      SHP(3,16) = -S2*(ONE+R)                                      ...02270
      SHP(4,16) = S2 *(ONE+R)*(ONE-T)                              ...02271
108   IF(NEL.LT.17) GO TO 200                                      ...02272
      IF(IX(17).EQ.0) GO TO 109                                    ...02273
      SHP(1,17) = T2*(ONE+S)                                       ...02274
      SHP(2,17) = T2*(ONE+R)                                       ...02275
      SHP(3,17) = -T*PFIVE*(ONE+R)*(ONE+S)                         ...02276
      SHP(4,17) = T2*(ONE+R)*(ONE+S)                               ...02277
109   IF(NEL.LT.18) GO TO 200                                      ...02278
      IF(IX(18).EQ.0) GO TO 110                                    ...02279
      SHP(1,18) = -T2*(ONE+S)                                      ...02280
      SHP(2,18) = T2*(ONE-R)                                       ...02281
      SHP(3,18) = -T*PFIVE*(ONE-R)*(ONE+S)                         ...02282
      SHP(4,18) = T2*(ONE-R)*(ONE+S)                               ...02283
110   IF(NEL.LT.19) GO TO 200                                      ...02284
      IF(IX(19).EQ.0) GO TO 111                                    ...02285
      SHP(1,19) = -T2*(ONE-S)                                      ...02286
      SHP(2,19) = -T2*(ONE-R)                                      ...02287
      SHP(3,19) = -T*PFIVE*(ONE-R)*(ONE-S)                         ...02288
```

```
      SHP(4,19) = T2*(ONE-R)*(ONE-S)
111   IF(NEL.LT.20) GO TO 200
      IF(IX(20).EQ.0) GO TO 112
      SHP(1,20) = T2*(ONE-S)
      SHP(2,20) = -T2*(ONE+R)
      SHP(3,20) = -T*PFIVE*(ONE+R)*(ONE-S)
      SHP(4,20) = T2*(ONE+R)*(ONE-S)
112   IF(NEL.LT.21) GO TO 200
C
C.... INTERIOR NODE (LAGRANGIAN)
      IF(IX(21).EQ.0) GO TO 200
      SHP(1,21) = -32.0D0*R*S2*T2
      SHP(2,21) = -32.0D0*S*R2*T2
      SHP(3,21) = -32.0D0*T*R2*T2
      SHP(4,21) = 64.0D0*R2*S2*T2
C
C.... CORRECT EDGE NODES FOR INTERIOR NODE (LAGRANGIAN)
      DO 114 J=1,4
      DO 113 I=1,8
113   SHP(J,I) = SHP(J,I) - 0.125D0*SHP(J,21)
      DO 114 I=9,20
114   IF(IX(I).NE.0) SHP(J,I) = SHP(J,I) - 0.25D0*SHP(J,21)
C
C.... CORRECT CORNER NODES FOR PRESENCE OF MIDSIDE NODES
200   K = 12
      DO 202 I = 1,4
      I4 = I + 4
      K4 = K + 4
      L = I + 8
      L4 = L + 4
      M = L+ 8
      DO 201 J = 1,4
      SHP(J,I ) = SHP(J,I ) - PFIVE*(SHP(J,K )+SHP(J,L )+SHP(J,M))
201   SHP(J,I4) = SHP(J,I4) - PFIVE*(SHP(J,K4)+SHP(J,L4)+SHP(J,M))
202   K=L
      RETURN
      END


      SUBROUTINE PGAUSS(GP,WF,NGP,L)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... GAUSS POINTS AND WEIGHTS
C
      DIMENSION GPWF(7,6)
      DATA GPWF /0.0D0,2.0D0,1.0D0,.555555555555556D0,.347854845137454D0
     1,.236926885056189D0,.171324492379170D0,.577350269189626D0,-.57735C
     2269189626D0,1.0D0,.888888888888889D0,.652145154862546D0,.4786286
```

```
     3499366CD0,.36076157304813900,.77459666924148300,0.0D0,-.77459666924...
     414830D0,.5555555555555556D0,.6521451548625460D0,.5688888888888889D0,.4...
     5679139345726910D0,.8611363115940530D0,.3399810435848560D0,-.3399810433...
     6584856D0,-.8611363115940530D0,.3478548451374540D0,.4786286704993660C...
     7,.4679139345726910D0,.9061798459386640D0,.5384693101056830D0,0.0D0,-...
     8538469310105683D0,-.9061798459386640D0,.2369268850561890D0,.36076157...
     9304813900,.9324695142031520D0,.6612093864662650D0,.2386191860831970C...
    1,-.2386191860831970C,-.6612093864662650D0,-.9324695142031520D0,.1713...
    2244923791700D0/
      NWF = NGP + 1
      GP = GPWF(L,NGP)
      WF = GPWF(NWF,L)
      RETURN
      END


      SUBROUTINE BCOND3 (UL,XL,IX,S,P,NDF,NDM,NST,NSBC,NGP,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      COMMON /ELDATA/ DM,N,MA,MCT,IEL,NEL
      DIMENSION UL(1),XL(NDM,1),IX(1),S(NST,1),P(1),SHP(4,21)
      DIMENSION KEL(50),KBCOND(50),KSURF(50),PROPB(50,2)
      GO TO (1,2),IFLAG
C
C.... INPUT SURFACE BOUNDARY CONDITIONS
1     WRITE(6,2000) O,HEAD
      KBMAX = 0
      DO 10 I=1,NSBC
      READ (5,1000) KEL(I),KBCOND(I),KSURF(I),PROPB(I,1),PROPB(I,2)
      KBMAX = MAX0(KBMAX,KBCOND(I))
10    WRITE(6,2001) KEL(I),KBCOND(I),KSURF(I),PROPB(I,1),PROPB(I,2)
      IF(KBMAX.GE.4)CALL CONV (T,COEFF,1)
      RETURN
C
C.... ADD SURFACE BOUNDARY CONDITIONS CONTIBUITIONS
C.... TO CONDUCTIVITY MATRIX AND LOAD VECTOR
2     DO 30 LL=1,NSBC
      IF(KEL(LL).NE.N) GO TO 30
      KBT = KBCOND(LL)
      KBS = KSURF(LL)
      CONST = PROPB(LL,1)
      TEMP =PROPB(LL,2)
      KBSABS = IABS(KBS)
      C3 = DFLOAT(KBS/KBSABS)
      DO 30 II=1,NGP
      CALL PGAUSS (C1,W1,NGP,II)
      DO 30 JJ=1,NGP
      CALL PGAUSS (C2,W2,NGP,JJ)
```

```
      WG = W1*W2
C
C.... SELECT CORRECT INTEGRATION POINTS
      GO TO (21,22,23),KBSABS
21    RG = C3
      SG = C1
      TG = C2
      GO TO 24
22    RG = C1
      SG = C3
      TG = C2
      GO TO 24
23    RG = C1
      SG = C2
      TG = C3
24    CALL SHAP3D (RG,SG,TG,XL,SHP,DUM,NDM,NEL,IX,.TRUE.)
      CALL JACBB3 (XSJ,KBSABS)
      DS = WG*XSJ
      COEFF = CONST
      GO TO (25,26,27,28),KBT
C
C.... FLUX BOUNDARY CONDITION
25    DO 251 L=1,NEL
251   P(L) = P(L) - COEFF*SHP(4,L)*DS
      GO TO 30
C
C.... RADIATION BOUNDARY CONDITION
27    T = 0.000
      DO 271 I=1,NEL
271   T = T + SHP(4,I)*UL(I)
      COEFF = COEFF*(T*T + TEMP*TEMP)*(T + TEMP)
      GO TO 26
C
C.... CONVECTION BOUNDARY CONDITION (TEMP. DEPENDENT)
28    T = 0.000
      DO 281 I=1,NEL
281   T = T + SHP(4,I)*UL(I)
      CALL CONV(T,COEFF,2)
C
C.... CONVECTION BOUNDARY CONDITION
26    DO 261 J=1,NEL
      A = SHP(4,J)*DS
      P(J) = P(J) + COEFF*TEMP*A
      DO 261 I = 1,NEL
261   S(I,J) = S(I,J) + COEFF*SHP(4,I)*A
30    CONTINUE
      RETURN
1000  FORMAT (3I5,2F10.0)
```

```
2000  FORMAT (A1,20A4,//5X,12HSURFACE B.C.//,
     15X,52H ELEM     B.C.   SURF.   PROPERTY VALUE       TEMPERATURE)
2001  FORMAT (5X,3(I5,2X),2(1PG15.7,2X))
      END


      SUBROUTINE PKZ (T,C,IFLAG)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... CALCULATE CONDUCTIVITY (Z DIRECTION)
C
      COMMON /CDATA/ O,HEAD(20),NUMNP,NUMEL,NUMMAT,NEN,NEQ,IPR
      DIMENSION TEMP(50),COEF(50)
      IF(IFLAG.EQ.1) WRITE (6,2000) O,HEAD
      CALL TABLE (TEMP,COEF,T,C,NDATA,IFLAG)
      RETURN
2000  FORMAT(A1,20A4,//5X,26HCONDUCTIVITY DATA (Z DIR.)//,
     1 5X,30H   TEMPERATURE              KY     )
      END


      SUBROUTINE JACBB3 (XSJ,KS)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C.... COMPUTE SURFACE JACOBIAN DETERMINANT
C
      COMMON /JAC3S/ XS
      DIMENSION XS(3,3)
      GO TO (1,2,3),KS
1     D1 = XS(2,2)*XS(3,3) - XS(2,3)*XS(3,2)
      D2 = XS(2,3)*XS(3,1) - XS(2,1)*XS(3,3)
      D3 = XS(2,1)*XS(3,2) - XS(2,2)*XS(3,1)
      GO TO 10
2     D1 = XS(1,1)*XS(3,3) - XS(1,3)*XS(3,1)
      D2 = XS(1,3)*XS(3,2) - XS(1,2)*XS(3,3)
      D3 = XS(1,2)*XS(3,1) - XS(1,1)*XS(3,2)
      GO TO 10
3     D1 = XS(1,1)*XS(2,2) - XS(1,2)*XS(2,1)
      D2 = XS(1,2)*XS(2,3) - XS(1,3)*XS(2,2)
      D3 = XS(1,3)*XS(2,1) - XS(1,1)*XS(2,3)
10    XSJ = DSQRT(D1*D1 + D2*D2 + D3*D3)
      RETURN
      END
```

LIST OF REFERENCES

1. Arpaci, V. S., Conduction Heat Transfer, Addison-Wesley Publishing Company, 1966.

2. Zienkiewicz, O. C. and Parekh, C. J., "Transient Field Problems: Two-Dimensional and Three-Dimensional Analysis by Isoparametric Finite Elements," International Journal for Numerical Methods in Engineering, v. 2, p. 61-71, January-March 1970.

3. Lew, G. T., A Three Dimensional Solution of the Transient Field Problem Using Isoparametric Finite Elements, M.S.M.E. Thesis, Naval Postgraduate School, Monterey, 1972.

4. Zienkiewicz, O. C., The Finite Element Method, 3rd ed., McGraw-Hill, 1977.

5. Bathe, K. J. and Wilson, E. L., Numerical Methods in Finite Element Analysis, Prentice-Hall, 1976.

6. Liniger, W., "Optimization of a numerical integration method for stiff systems of ordinary differential equations," IBM Research Report RC2198, 1968.

7. Wood, W. L., "On the Zienkiewicz Three- and Two-Time-Level Schemes Applied to the Integration of Parabolic Equations," International Journal for Numerical Methods in Engineering, v. 12, p. 1717-1726, 1978.

8. Lees, M., "A Linear Three-Level Difference Scheme for Quasilinear Parabolic Equations," Mathematics of Computation, v. 20, p. 516-622, 1966.

9. Hogge, M. A., A Survey of Direct Integration Procedures for Nonlinear Transient Heat Transfer, presented at the International Conference on Numerical Methods in Thermal Problems, University College, Swansea, U. K., 2-6 July, 1979.

10. Wood, W. L. and Lewis R. W., "A Comparision of Time Marching Schemes for the Transient Heat Conduction Equation," International Journal for Numerical Methods in Engineering, v. 9, p. 679-689, 1975.

11. Gresho, P. M. and Lee, R. L., Don't Supress the Wiggles - They're Telling You Something!, presented at the

Symposium on Finite Element Methods for Convection Dominated Flows, ASME Winter Annual Meeting, New York, 2-7 December, 1979.

12. Haji-Sheikh, A. and Sparrow, E. M., "The Solution of Heat Conduction Problems by Probability Methods," Journal of Heat Transfer, v. 89, p. 121-131, May 1967.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
4.	Professor Gilles Cantin, Code 69Ci Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	10
5.	Professor O. C. Zienkiewicz, Code 69Zw Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
6.	1º Ten. Jorge Bettencourt Direcção Do Serviço De Instrução Ministério Da Marinha Lisboa, PORTUGAL	3
7.	Professor Paul F. Pucci, Code 69Pc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
8.	Associate Professor Matthew D. Kelleher Code 69Kk Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
9.	Associate Professor David Salinas, Code 69Zc Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93940	1
10.	Associate Professor Richard H. Franke Code 53Fe Department of Mathematics Naval Postgraduate School Monterey, California 93940	1

11. Professor R. E. Newton, Code 69Ne 1
Department of Mechanical Engineering
Naval Postgraduate School
Monterey, California 93940

12. Professor A. P. Boresi, Code 69Bh 1
Department of Mechanical Engineering
Naval Postgraduate School
Monterey, California 93940

13. LCDR Lael R. Easterling, USN 1
4243 N.W. 54th St.
Oklahoma City, Oklahoma 73112

14. Georges Verchery 1
Départment de Génie Mécanique
Université de Technologie
60200 Compiègne, FRANCE

15. Jean Louis Armand 1
Institut de Recherche Pour la
Construction Navale
3 Avenue de Grand Champ
78230 le Pecq, FRANCE

16. Prof. K. J. Bathe 1
Mechanical Engineering Department
M.I.T.
77 Massachusetts Avenue
Cambridge, Massachusetts 02139

17. William J. Dodge 1
Oak Ridge National Laboratory
Building 9204-1 Box Y
Oak Ridge, Tennessee 36830

18. Jack Tree 1
Air Research Manufacturing Co.
402 South 36th Street
P.O. Box 5217
Phoenix, Arizona 85010

19. Prof. Edward L. Wilson 1
Structural Engineering Division
Civil Engineering Department
University of California (Berkeley)
Berkeley, California 94720

20. Dr. William J. Stronge (Code 603) 1
Naval Weapons Center
China Lake, California 93955

21. J. E. Serpanos 1
Code 3162
Naval Weapons Center
China Lake, California 93955

22. Dr. Jean Louis Batoz 1
Département de Génie Mécanique,
Université de Technologie,
60200 Compiègne, FRANCE

23. Professor Guri Dhatt 1
Centre Technique de l'Informatique
Université Laval
Québec, Prov. de Québec
CANADA GIK 7P4

24. John Fairbanks 1
Department of Energy
Division of Power Systems
20 Massachusetts Avenue, NW
Washington, DC 20545

25. Dr. Gilbert Touzot 1
Centre d'Informatique
Université de Technologie
60206 Compiègne, FRANCE

26. R. A. Langworthy 1
Applied Technology Laboratories
U.S. Army Research and Technology
Laboratory
Fort Eustis, Virginia 23604

27. E. M. Lenoe 1
Army Materials & Mechanic Research
Center
Arsenal Street
Watertown, Massachusetts 02172

28. Dr. Paris Genalis 1
Naval Ship Research and Development Center
Bethesda, Maryland 20084

29. C. Miller 1
Naval Sea Systems Command
Department of the Navy
Washington, DC 20362

30. Code SEC 6734 1
Naval Ship Engineering Center
Philadelphia Division
Philadelphia, Pennsylvania 19112

31. A. M. Diness (Code 471) 1
Department of the Navy
Office of Naval Research
Arlington, Virginia 22217

32. Ray M. Standahar 1
Office of Secretary of Defense
DDR&E
3D1089 Pentagon
Washington, DC 20301

33. R. Rice (Code 6360) 1
Naval Research Laboratory
Washington, DC 20375

34. E. Van Reuth 1
Defense Advanced Research Projects Agency
1440 Wilson Boulevard
Arlington, Virginia 22209

35. I. Machlin 1
Naval Air System Command
Department of the Navy
Washington, DC 20361

36. B. Probst, MS 49-3 1
NASA-Lewis Research Center
21000 Brookpark Road
Cleveland, Ohio 44135

37. Mr. C. P. Blankenship, MS 105-1 1
NASA-Lewis Research Center
21000 Brookpark Road
Cleveland, Ohio 44135

38. Dr. H. Graham/AFML/LLM 1
Department of the Air Force
Air Force Materials Laboratory
Wright-Patterson Air Force Base, Ohio 45433

39. Mr. George Strong 1
DCASMA, Phoenix
3800 North Central Avenue
Phoenix, Arizona 85012

40. S. Freiman (Code 6363) 1
Naval Research Laboratory
Washington, DC 20375

41. AFML/LLM/N. M. Geyer 1
Air Force Materials Laboratory
Wright-Patterson Air Force Base, Ohio 45433

42. S. Wiederhorn 1
National Bureau of Standards
Washington, DC 20234

43. LT J. H. Preisel, Jr., USN 1
922 Bernard Road
Peekskill, New York 10566

44. Allan F. Greiner 1
United Technologies Research Center
East Hartford, Connecticut 06108

45. LCDR L. B. Elliott 1
Code 338.14
Long Beach Naval Shipyard
Long Beach, California 90822

46. Martin A. Krenzke 1
David W. Taylor Naval Ship
Research and Development Center
A236, Bldg. 19
Carderock Laboratory
Bethesda, Maryland 20084

47. Professor Robert L. Taylor 1
Department of Civil Engineering
University of California
at Berkeley
Berkeley, California 94720

Thesis 188011
B4589 Bettencourt
c.1
Finite element analysis program (FEAP) for conduction heat transfer.